世界数学名题欣赏丛书

素数判定与大数分解

孙琦
旷京华
编著

辽宁教育出版社
1987年·沈阳

素数判定与大数分解

孙琦　旷京华　编著

辽宁教育出版社出版　辽宁省新华书店发行
（沈阳市南京街6段1里2号）　大连印刷工业总厂印刷

字数：52,000　开本：787×1092 1/32　印张：3 1/4　插页：4
印数：1—4,086
1987年10月第1版　1987年10月第1次印刷

责任编辑：俞晓群　谭坚　责任校对：周广东
封面设计：安今生　插　图：安　迪

统一书号：7371·506　定价：1.00元
ISBN 7—5382—0180—7

内 容 简 介

本书是”世界数学名题欣赏丛书“之一。素数判定与大数分解问题在数论中占有重要地位，远古时代人们就十分重视它的研究。近年来，由于计算机科学的发展，使这一古老的问题焕发了青春，形成了数论中的新分支——计算数论。本书完整地介绍了素数判定问题的全部历史和理论，阐明了它在纯数学研究和应用数学研究中的地位，及其在当代科学中的实用价值(如在密码学中的作用)。全书内容丰富，论述严整。

Summary

This book is one of a Series of World Famous Mathematics Appreciat ion. Decidability of prime number and the problem of decomposition of large numbers hold an important place in number theory. From ancient times people paid great attention to its research. Because of the development of computer science recently the old problem forms a new branch of number theory —— computation of number theory. The book completely introduces the whole history and theories of the problem of decidability of prime number, and expounds its position in the research of pure mathematics, applied mathematics, and practical value in modern science (e.g. the use in Cryptography). The book has subst antial content and the exposition is inneat formation.

序　言

数论中一个最基本、最古老而当前仍然受到人们重视的问题就是判别给定的整数是否素数（简称为素数判别或素性判别）和将大合数分解成素因子乘积（简称为大数分解）。在历史上，这个问题曾经吸引了包括费马（Fermat）、欧拉（Euler）、勒让德（Legendre）和高斯（Gauss）在内的大批数学家，他们花费了大量的时间和精力去研究这个问题。高斯在其著名的《算术探讨》（《Disquisitiones Arithmeticae》）中称道："把素数同合数鉴别开来及将合数分解成素因子乘积被认作为算术中最重要最有用的问题之一。"我国的《易经》中也对这个问题作了研究。

素数判别和大数分解这个问题具有很大的理论价值。因为素数在数论中占有特殊的地位，鉴别它们则成为最基本的问题；而把合数分解成素因子的乘积是算术基本定理的构造性方面的需要。人类总是有兴趣问如下的问题：$2^{131}-1$ 是否素数？由23个1组成的数是否素数？怎么分解

314876948415723617对素数判别和大数分解的研究必然会丰富人类的精神财富。更重要的是，素数判别和大数分解具有很大的应用价值。在编码中，需要讨论某类有限域及其上的多项式，这类有限域就是由素数p所作成的$Z/pZ=\{\overline{0},\overline{1},\cdots,\overline{p-1}\}$，这就要求我们去寻找素数、判别素数。在快速数论变换中，要讨论 Z/nZ 上的卷积运算；就要知道 Z/nZ 的乘法群的构造，而这就依赖于将 n 分解成素因子的乘积。下面介绍的 RSA 公开密钥码体制更加说明了这个问题的两个方面在实际应用中的作用。1977 年，艾德利曼（Adleman）、希爱默（Shamir）和鲁梅利（Rumely）发明了一个公开密钥码体制。在这个密码体制中，对电文的加密过程是公开的，但是，你仅知道加密过程而未被告知解密过程则不可能对电文进行解密。他们的体制就是依靠这样一个事实：我们能够很容易地将两个大素数（譬如两个百位素数）乘起来；反过来，要分解一不大整数（譬如200位）则几乎不可能。（关于 RSA体制的详细介绍，请参阅文献〔1〕）。因此 RSA体制就与素数判别和大数分解有密切联系。首先，要具体建立一个 RSA 体制就需要两个大素数，因而就涉及到寻找大素数的问题；而RSA体制的破译之可能性就依赖于分解一个大数可能性。于

是，RSA体制的建立与破译就等价于素数判别与大数分解问题。近来，由于计算机科学的发展，人们对许多数学分支的理论体系重新用计算的观点来讨论。从计算的观点来讨论数论问题形成了当前很活跃的分支——计算数论。而素数判别和大数分解成为这一分支的重要组成部分。在这一部分里提出了两个重要的、悬而未决的问题：是否存在判别素数的多项式算法？是否存在分解大整数的多项式算法？已知道"分解整数"这个问题是一个NP完全问题，因此对上面第二个问题的讨论是解决计算机科学中的难题*："NP完全问题是否一定是多项式算法可解的？"的一个突破口。因此，素数判别和大数分解对计算机科学来说也是很有价值的。

最直接的素数判别和大数分解方法就是试除法，即对整数 n，用 2，…，$n-1$ 去试除，来判定 n 是否素数，分解式如何。这个方法是最简单的一个方法，古希腊时就被人所知，但这个方法对较大的数（20位左右）就要耗费很多时间。在本世纪四十年代电子计算机出现之前，尽管产生了许多素数判别和大数分解方法，但因为用手算，速度太慢，很多方法在实用中即使对十几位

* 可参看：管梅谷，组合最优化介绍，数理化信息，1，73—80。

的数也需要好几天，而对更大的数就无能为力了。随着计算机的出现及发展，人们开始用这个有力的工具来研究素数判别和大数分解。到六十年代末期，已产生了许多新方法，历史上的许多方法也得到了应用，使得对四十几位数的素数判别可以很快得到结果。而到七十年代末，数论学家和计算机专家们已深入地研究了这个问题，得到许多实际有效的方法。用这些方法在较好的计算机上判别一个100位数是否素数只需不到一分钟；分解70位左右的整数也是日常工作了。这些成果已引起人们的普遍关注。在这个领域中的研究空前活跃。虽然离问题的彻底解决还很远，但在本领域中已取得了一个又一个的突破。在这方面的研究必有光辉的前景。

我们写这个小册子的目的是要介绍素数判别和大数分解的发展历史、一般理论、各种方法及最新成果，是想让许多非专业的读者了解这个方向的内容和进展情况。当然，只有在这些定理的证明较为初等而又不太长时，我们才给出其证明。因为这个方向与计算机科学的密切关系，我们还要结合计算量来介绍一些数论中常用的基本算法。

除了极个别内容，如第二章第七节，本书的绝大部分内容只需要某些初等数论的知识，它们

可以在任何一本介绍初等数论的书中都能找到，如文献〔1〕。对于广义黎曼猜想，我们写了一则简短的附录。作为“世界数学名题欣赏丛书”中的一本，如果读者在欣赏之余，还打算进一步学习和探讨的话，那么，后面所列的文章和书目，可供参考。

限于水平，本书的缺点和错误一定不少，我们期待着读者的批评指正。

作　者

1987年4月





作者简介

孙琦（左），1937年生于浙江省吴兴县，1961年毕业于四川大学数学系，现为四川大学数学系教授、数学研究所数论研究室主任、《数学学报》编委、四川省数学会副秘书长。已发表学术论文50余篇，出版著作五种。主要研究方向为数论中的不定方程和应用数论。

旷京华，1963年生于江西，1982年毕业于江西大学数学系，继续读研究生于四川大学数学系，现在美国明尼苏达大学数学系攻读数学。主要研究方向是与数论有关的分支。

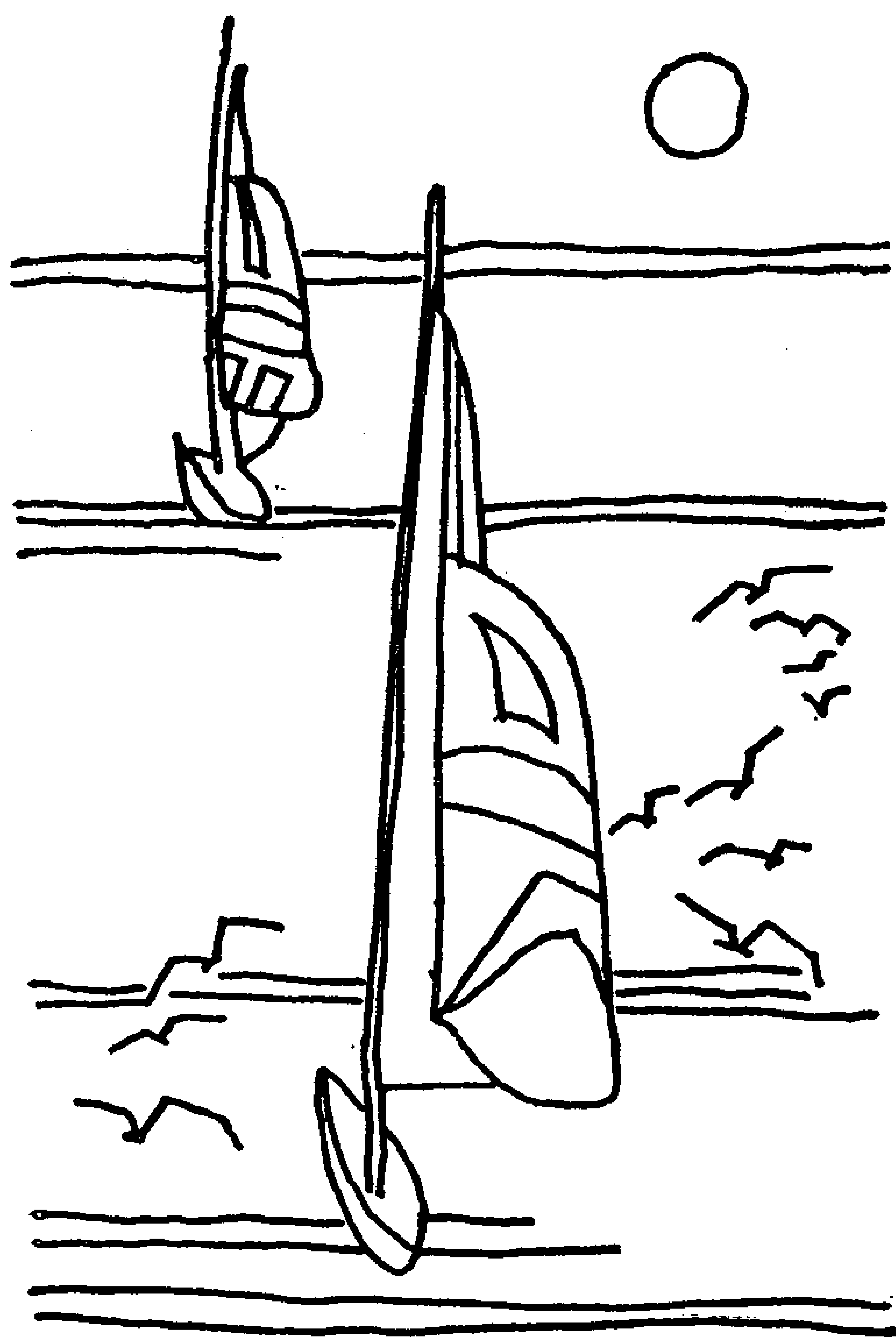
世界数学名题欣赏丛书

费马猜想
黎曼猜想
连续统假设
希尔伯特第十问题
欧几里得第五公设
哥德尔不完全性定理
不动点定理
无处可微的连续函数
科克曼女生问题
斐波那契数列
哥德巴赫猜想
置换多项式及其应用
素数判定与大数分解
货郎担问题

目　录

Catalogue

一　数论中的基本算法



1. 算法及其计算量的概念

通常，解决问题的方式有两种。其一是对问题的每个对象（也称作输入），直接给出答案，其二是给出一套规则，使得对问题的任何一个对象（输入），解答者可依照这些规则、机械地执行运算，能在有限步内得到答案。这里的第二种方式就是问题的算法解答，这时也称问题是算法可解的，而所给出的规则就称为算法。

例 在复数范围内解一元二次方程。

这个问题的输入即是具体给出的一元二次方程。任给一个方程$ax^2+bx+c=0$（$a\neq 0$），则方程的两个解是$\frac{-b\pm\sqrt{b^2-4ac}}{2a}$。这就是第一种解答方式。

例 判别任意给出的自然数是否素数。

熟知，我们没有象上例那样的公式来表明哪个数是素数，哪个数不是素数。但是，我们可以以如下方式解答此问题：对任意给定的数n，用2，

3,…,n－1 去试除n，若其中有一个除尽n，则n不是素数；否则，n 是素数。这就是问题的算法解答。

注意，算法不可解的问题是存在的。希尔伯特的第十问题* 就是一个算法不可解的问题。这一点是马递佳塞维奇于1970年证明的。

对算法可解的问题而言，解答的算法可能很多，我们在实际解决问题时究竟采取哪一种算法呢？这就要求对算法的好坏进行评判。算法的好坏与所谓的计算量有密切关系。人们注意到，对某类问题的解答依赖于一些基本运算。譬如，在排序问题中，比较运算——即比较两个元的先后——是一种基本运算。当然，“互换位置”也可以当作是一种基本运算。一个算法在对问题的某个输入解答所执行的基本运算次数，称为算法对此输入执行的计算量。算法对不同的输入执行的计算量一般不同。在把计算量描述为输入的函数之前，需要对输入给个度量，这就是所谓的输入尺寸。例如，在排序问题中，输入尺寸可定义为待排序的贯的长度。这样一来，计算量可以描述为输入尺寸的函数。算法对问题的所有输入执行

---

* 希尔伯特第十问题："判定任何一个不定方程 $f(x_1,\cdots,x_n)=0$（其中 $f(x_1,\cdots,x_n)$ 是多元整系数多项式）有否解？"这是1900年，希尔伯特在国际数学家大会上提出来的。

的计算量的最大值称为算法在最坏性态下的计算量。如果解决一个问题有两个算法，其中一个算法在最坏性态下的计算量比另一个在最坏性态的计算量少，则称前者在最坏性态下比后者好。若一个问题存在一个算法解，其算法在最坏性态下的计算量是输入尺寸的多项式函数，则称此问题存在多项式算法，也说问题是P问题，否则称它不存在多项式算法。

最后，我们谈谈概率算法和确定算法。所谓概率算法就是它的某些步骤是要依靠服从某种分布的随机抽样来完成的，这种随机过程应是有限步内可完成的，而且算法得出的结论应与所作的随机抽样无关，利用随机手段仅仅是为了加快算法的进程或为了方便。确定算法是相对于概率算法而言的，即它的每一步骤都是确定的，不须用随机手段就可完成的。以后，这两种算法都会遇上，凡没有用随机手段的都是确定算法，我们将不作特别说明了。

2. 数论中的基本算法

在数论问题中，输入一般是一个或几个自然数。如果输入是一个自然数 n，则定义其输入尺寸为它的二进位表示的位数，即 $[\log_2 n]+1$，有

时也将 $\log_2 n$ 作为输入尺寸。熟知，数论问题的解答中，自然数的加减乘除这四则运算是基本的运算，而两个个位数的加法、减法和乘法及两位数除以个位数的除法又最基本。因此，我们如下定义数论问题的基本运算：假如我们是在 r- 进位制中讨论自然数的运算（通常在十进位下讨论，即 $r=10$；而在计算机上，一般在二进位下讨论，即 $r=2$），则基本运算就是个位数的相加、相减、相乘，两位数除以个位数的除法及向左移位运算（即乘上 r）。有了基本运算，就可以来讨论数论中的基本算法了。

（1） 四则运算

加法：回忆一下小学里学多位数相加的情景，当时是列竖式再按老师教的规则去演算。这些竖式规则就是算法。我们依照用竖式演算的步骤将其用文字写出来即是：任意给定两个 n 位的 r 进位数（如有一个没够n位，可添一些零而达到 n 位），$a=(a_{n-1}\cdots a_1a_0)_r=a_{n-1}r^{n-1}+a_{n-2}r^{n-2}+\cdots+a_1r+a_0$，$b=(b_{n-1}\cdots b_1b_0)_r=b_{n-1}r^{n-1}+b_{n-2}r^{n-2}+\cdots+b_1r+b_0$，则 $a+b=\sum_{i=0}^{n-1}(a_i+b_i)r^i$。现在要来求 $a+b$ 的 r 进位表示。首先，$a_0+b_0=c_0r+s_0$，其中 $0\leqslant s_0<r$，$c_0=0$ 或 1 视 $a_0+b_0<r$ 或 $\geqslant r$ 而定。接着有 $a_1+b_1+c_0=c_1r+s_1$，$0\leqslant s_1<r$，因为 $0\leqslant c_0\leqslant 1$，$0\leqslant a_1$，$b_1\leqslant r-1$，

故 $c_1 = 0$ 或 1 视 $a_1 + b_1 + c_0 < r$ 或 $\geqslant r$ 而定，继续对第 2，3，…，$n-1$ 讨论得，存在 c_i, s_i 使 $a_i + b_i + c_{i-1} = c_i \cdot r + s_i$，其中 $0 \leqslant s_i < r$，$c_i = 0$ 或 1 $(i = 1, 2, \cdots, n-1)$，最后令 $c_{n-1} = s_n$，则

$$
\begin{aligned}
a + b &= \sum_{i=0}^{n-1}(a_i + b_i) r^i = \sum_{i=1}^{n-1} (c_i r + s_i - c_{i-1}) r^i + (c_0 r + s_0) \\
&= \sum_{i=0}^{n-1} s_i r^i + \sum_{i=1}^{n-1} c_i r^{i+1} - \sum_{i=1}^{n-1} c_{i-1} r^i + c_0 r \\
&= \sum_{i=0}^{n-1} s_i r^i + c_{n-1} r^n = \sum_{i=1}^{n} s_i r^i = (s_n s_{n-1} \cdots s_1 s_0)_r
\end{aligned}
$$

因为由 a_i，b_i，c_{i-1} 经带余除法 $a_i + b_i + c_{i-1} = c_i r + s_i$，$0 \leqslant s_i < r$，$c_i = 0$ 或 1 确定 c_i，s_i 至多需要 5 次基本运算，故用竖式算法计算两个 n 位数的计算量至多是 $5n$，用O记号即是 $O(n)$。在二进位制中，$r = 2$，输入n 的位数是$[\log_2 n] + 1$，另外，对任意的$r > 1$ 有 $[\log_r n] + 1 = O(\log_2 n)$，故得到下面的定理。

定理 1.1 用竖式算法计算两个不大于 n 的数相加时，其计算量是$O(\log_2 n)$。

减法：同加法一样地讨论，只是将 $a + b$、$a_i + b_i$ 等改成 $a - b$，$a_i - b_i$，相应地，$c_i = 0$ 或 -1 视 $a_i - b_i + c_{i-1} \geqslant 0$ 或 $\leqslant 0$ 而定。我们也可以得到下面的定理。

定理 1.2 用竖式算法计算两个不大于 n 的

数相减时，其计算量是$O(\log_2 n)$。

乘法：在用竖式算法作多位数的乘法时，先是做个位数与多位数相乘，然后再移位相加．于是，我们先来讨论个位数与多位数相乘．设 $a=(a_{n-1}\cdots a_1a_0)_r=\sum_{i=0}^{n-1}a_ir^i$，$b=(b_0)_r=b_0$，这时，$ab=\sum_{i=0}^{n-1}a_ib_0r^i$，设 $a_0b_0=c_0r+s_0$，$0\leqslant s_0<r$，因为 $0\leqslant a_0\leqslant r-1$，$0\leqslant b_0\leqslant r-1$，故 $0\leqslant c_0\leqslant r-2$，依次有 $a_ib_0+c_{i-1}=c_ir+s_i$，其中 $0\leqslant s_i<r$，因为 $0\leqslant a_i$，$b_0\leqslant r-1$，$0\leqslant c_{i-1}\leqslant r-2$，故 $0\leqslant a_ib_0+c_{i-1}\leqslant r(r-2)+(r-1)$，因而 $0\leqslant c_i\leqslant r-2$，以上 $i=1,\cdots,n-1$．再令 $s_n=c_{n-1}$，则得 $ab=\sum_{i=0}^{n}s_ir^i=(s_ns_{n-1}\cdots s_1s_0)_r$。

一般地，对两个多位数相乘，设 $a=(a_{n-1}\cdots a_1a_0)_r=\sum_{i=0}^{n-1}a_ir^i$，$b=(b_{m-1}\cdots b_1b_0)_r=\sum_{i=0}^{m-1}b_ir^i$，

则有
$$a\cdot b=\sum_{i=0}^{m-1}(a\cdot b_i)r^i \qquad (1)$$

因为$(a\cdot b_i)r^i$是$a\cdot b_i$的 r 进位表示式向左移 i 位，即在$a\cdot b_i$的r 进位表示式之后添 i 个零．因此表达式(1)表示 m个数移位相加，这就将多位数乘法归结为多位数与个位数相乘，然后再移位相加。

注意到，在个位数与多位数相乘时，由 a_i，b_i，c_{i-1} 经 $a_ib_i+c_{i-1}=c_ir+s_i$，$0\leqslant s_i<r$ 确定 c_i，s_i 至多需要 5 次基本运算，故个位数与 n 位数相

乘的计算量至多是5n，而（1）式表明，n位数与m位数相乘，需要作m次个位数与n位数相乘，作$\frac{m(m-1)}{2}$次左移运算及$n+1$位数与$n+2$位数、$n+2$位数与$n+3$位数、……、$n+m-1$位数与$n+m$位数相加各一次，故总的计算量不超过$5mn+\frac{m(m-1)}{2}+5n(m-1)+5\cdot\frac{m(m-1)}{2}\leqslant 13mn$（这里不妨设$m\leqslant n$）。因此，竖式算法做n位数与m位数（$m\leqslant n$）相乘的计算量不多于13mn。设M(n)表示两个n位数相乘的计算量，则$M(n)=O(n^2)$。故可得下面的定理。

定理 1.3 竖式算法做两个不超过n的数的相乘时，其计算量是$O(\log_2^2 n)$。

带余除法：带余除法的竖式算法要用文字表述出来比先前的加法和乘法都要麻烦，但它只不过是将小学里做带余除法的过程详细地写出来而已，这里不准备重复这些枯燥的叙述，而只将带余除法的竖式算法的计算量写出来。

定理 1.4 用竖式算法作两个不超过n的数的带余除法时，其计算量为$O(\log_2^2 n)$。

实际上，我们是得到这样的结论：一个2n位数除以一个n位数所得的商和余数的计算需要计算量$O(n^2)$，之后，才有定理 1.4。而且由此可

得，对一个不超过 m^2 的数 a 取模 m 得最小非剩余的计算量是 $O(\log^2_2 m)$。

关于自然数的乘法，我们还想介绍两个优美的结果，其证明已超出本书的范围。

定理 1.5 任给一个正数 ε，无论如何小，都存在一个乘法算法，它做两个 n 位数相乘的计算量是 $O(n^{1+\varepsilon})$，或者说，它做两个不大于 n 的数相乘的计算量是 $O(\log_2^{1+\varepsilon} n)$。

定理 1.6 存在乘法算法，使其作两个 n 位数相乘的计算量是 $O(n\cdot\log_2 n\cdot\log_2\log_2 n)$，或者说，它做两个不大于 n 的数相乘的计算量是

$$O(\log_2 n\cdot\log_2\log_2 n\cdot\log_2\log_2\log_2 n)$$

定理 1.6 中的乘法所需的计算量被认为是作乘法运算的最优计算量，即不存在有更少计算量的乘法算法。〔2〕

下面的定理表明，除法所需的计算量与乘法所需的计算量相当，它的证明也不在本书范围内，读者可参见〔2〕。

定理 1.7 存在除法算法，它求一个 2n 位数除以一个 n 位数得的商和余数所需的计算量是 $O[M(n)]$，其中 $M(n)$是做两个 n 位数乘法所需的计算量。

尽管定理 1.5 和定理 1.6 说明了有比竖式算法快得多的乘法算法，但为方便起见，我们以后

还是使用竖式算法的计算量来代表两个数相乘的计算量。

（2） 幂运算

给定一个整数 a，由 $a^k=a\cdot a^{k-1}$可以方便地归纳地定义了 a 的任意次幂。在素数判别和大数分解的讨论中，我们常常遇到要计算 $a^n \bmod m$。如果根据定义，先计算 $a \bmod m$，再依次计算 $a^k \bmod m=(a \bmod m)\cdot(a^{k-1} \bmod m) \bmod m$，则至少需要n次乘法才能得到 $a^n \bmod m$，而 n 经常是很大很大，这时计算量也就很大，于是需要有更好的计算$a^n \bmod m$的方法。我们确实有更好的方法，先来看一个例子。

例 计算 $3^{107} (\bmod 134)$

将107写成二进位数，即 $107=(1101011)_2$。我们有 $3^1\equiv3(\bmod 134)$，$3^2\equiv9(\bmod 134)$，$3^{2^2}\equiv81(\bmod 134)$，$3^{2^3}\equiv-5(\bmod 134)$，$3^{2^4}\equiv25(\bmod 134)$，$3^{2^5}\equiv-45(\bmod 134)$，$3^{2^6}\equiv15(\bmod 134)$。（以上的一串同余式中，每一个都是前一个的平方而得）。故$3^{107}=3^{2^6+2^5+2^3+2+1}=3^{2^6}\cdot3^{2^5}\cdot3^{2^3}\cdot3^2\cdot3^1\equiv15\cdot(-45)\cdot(-5)\cdot9\cdot3\equiv135\cdot135\cdot5\equiv5(\bmod 134)$。

上例中的方法可以推广到一般情况。当要计算 $a^n \bmod m$时，先将 n 用二进位制表示，设 $n=(a_{k-1}\cdots a_1 a_1)$，$a_i=0$ 或 1，$i=0, 1,\cdots,k-1$。

再计算 $r_0 = a \bmod m$，$r_1 = a^2 \bmod m$，…，$r_{k-2} = a^{2^{k-2}} \bmod m$，$r_{k-1} = a^{2^{k-1}} \bmod m$。再将使 $a_i = 1$ 对应的 r_i 连乘起来，取模m，则得 $a^k \bmod m$（这里连乘是指每乘一个数取一次模 m，然后用所得的结果再与另一乘数相乘）。我们有下面的定理

定理 1.8 给定 a，存在求幂取模的算法，使得其计算 $a^n \bmod m$ 的计算量为 $O(\log_2 n \log_2^2 m)$。

证明 在上面所描述的算法中，$r_0 = a \bmod m$。因为 a 是一个事先给定的常数，故 r_0 的计算可以忽略。而 $r_i = r^2_{i-1} \bmod m$，$i = 1, \cdots, n-1$，因而计算每个 r_i 的计算量是 $O(\log_2^2 m)$（其中，r_{i-1} 乘 r_{i-1} 的计算量为 $O(\log_2^2 r_{i-1}) \leqslant O(\log_2^2 m)$，取模 m 的计算量为 $O(\log_2^2 m)$），而这里 $k = [\log_2 n] + 1 = O(\log_2 n)$，故计算 r_0，r_1，…，r_{k-1} 的总共计算为 $O(\log_2 n \cdot \log_2^2 m)$。在把对应于 $a_i = 1$ 的 r_i 连乘起来时，每作一次乘法取一次模 m，在这串连乘积中，乘法次数至多 $[\log_2 n] + 1$ 次，取模m的次数也至多 $[\log_2 n + 1]$ 次，而且每次乘法的两个乘数都不超过 m，故做连乘积的计算量是 $O(\log_2 n \log_2^2 m)$。因此，计算 $a^n \bmod m$ 总共需要计算量 $O(\log_2 n \log_2^2 m)$。□

（3） 努卡斯序列的项的计算.

所谓努卡斯序列是指

$$U_n = \frac{\alpha^n - \beta^n}{\alpha - \beta}, \quad V_n = \alpha^n + \beta^n, \quad n = 0, 1, \cdots$$

其中 α 和 β 是以下整系数二次方程的根：

$$x^2 - Px + Q = 0, \quad (P, Q) = 1$$

在以后讨论素数判别时，我们需要计算努卡斯序列的第 n 项取模m：$U_n \bmod m$ 和 $V_n \bmod m$。它们的计算可利用努卡斯序列的性质，象讨论 $a^n \bmod m$ 的计算一样，得到比较好的算法，因为讨论它的基本思路与上一段一样，且要用到努卡斯序列的一些较繁杂的性质。这里，我们只叙述一个结果，读者有兴趣可自己证明这个结果。

定理 1.9 给定 P、Q，$(P,Q)=1$，存在算法使其计算 $U_n \bmod m$ 和 $V_n \bmod m$ 的计算量是 $O(\log_2 n \log_2^2 m)$。

（4） 进位制表示的互化

我们常常需要将一个自然数的一种进位制表示化为另一种进位制表示。譬如，日常给出的自然数是十进位数，要将它输到计算机中就必须将它化成二进位数，反过来，从计算机输出的结果又要从二进位数化为十进位数，才能使常人看懂结果。

设自然数n，由 r_1 进位制给出，要将它化为 r_2 进位制，我们可以如下完成：在 r_1 进位制体系中进行四则运算，特别是带余除法运算，有

$$
\left\{
\begin{array}{ll}
n = n_0 = n_1 r_2 + q_0, & 0 \leqslant q_0 < r_2 \\
n_1 = n_2 r_2 + q_1, & 0 \leqslant q_1 < r_2 \\
\vdots & \\
n_{k-2} = n_{k-1} r_2 + q_{k-2}, & 0 \leqslant q_{k-2} < r_2 \\
n_{k-1} = q_{k-1}, & 0 \leqslant q_{k-1} < r_2
\end{array}
\right. \qquad (2)
$$

则$(q_{k-1}q_{k-2}\cdots q_1\, q_0)_{r_2}$ 便是 n 在 r_2 进位制中的表达式。由带余除法的计算量（见定理 1.4）知，完成(2)的计算量是$O(\log_2{}^3 n)$。

（5） 最大公因数的算法

求最大公因数的最普遍的算法是欧几里得算法，它最初是公元前由欧几里得提出来的，有时也称它为辗转相除法。表述如下：

设给定$m, n(m>n)$，令 $r_0 = m$，$r_1 = n$，有

$$
\left\{
\begin{array}{ll}
r_0 = r_1 q_1 + r_2, & 0 \leqslant r_2 < r_1 \\
r_1 = r_2 q_2 + r_3, & 0 \leqslant r_3 < r_2 \\
\vdots & \\
r_{k-2} = q_{k-1} r_{k-1} + r_k, & 0 \leqslant r_k < r_{k-1} \\
r_{k-1} = q_k r_k &
\end{array}
\right. \qquad (3)
$$

则得 $r_k = \gcd(r_{k-1}, r_k) = \gcd(r_{k-2}, r_{k-1}) = \cdots = \gcd(r_2, r_3) = \gcd(r_1, r_2) = \gcd(r_0, r_1) = \gcd(m, n)$。

欧几里得算法(3)中作带余除法的次数k 可

由m和n确定出来。我们介绍下面的定理。

定理1.10 (3)式中的 $k<5\log_{2}n+1$，即得辗转相除的次数不大于n的十进位表示的位数的5倍。

证明 引入斐波那契序列 $F_0=0$，$F_1=1$，$F_n=F_{n-1}+F_{n-2}$，$n=2,3,4,\cdots$，易知有 $F_n=\frac{1}{\sqrt{5}}\cdot\left[\left(\frac{1+\sqrt{5}}{2}\right)^n-\left(\frac{1-\sqrt{5}}{2}\right)^n\right]$. 由(3)式得，$r_k\geqslant 1=F_2$，$r_{k-1}>r_k$ 故 $r_{k-1}\geqslant r_k+1\geqslant 2=F_3$，$r_{k-2}\geqslant r_{k-1}+r_k\geqslant F_2+F_3=F_4$，……，$r_1\geqslant r_2+r_3\geqslant F_k+F_{k-1}=F_{k+1}$。而 $F_{k+1}>\left(\frac{1+\sqrt{5}}{2}\right)^{k-1}$ 当 $k\geqslant 2$ 时成立，故 $n=r_1\geqslant F_{k+1}>\left(\frac{1+\sqrt{5}}{2}\right)^{k-1}$，即 $\log_{10}n>(k-1)\log_{10}\left(\frac{1+\sqrt{5}}{2}\right)$，而 $\log_{10}\frac{1+\sqrt{5}}{2}>\frac{1}{5}$，故 $\log_{10}n>\frac{1}{5}(k-1)$，因而 $k<5\log_{10}n+1$。设n的十进位数表示的位数是 l，则有 $n<10^l$，故 $k<5l+1$，而 k和$5l+1$都是整数，故$k\leqslant 5l$。□

推论 用欧几里得算法求m，n$(m\geqslant n)$的最大公因数的计算量是 $O(\log_2{}^3m)$。

证明 因为(3)式中的 $k<5\log_{10}n+1$，而且，其中每次带余除法的被除数和除数都不大于

m，故由定理 1.4 知，每次带余除法的计算量是 $O(\log_2^2 m)$，再由定理 1.10 及 $n \leqslant m$ 即得，由（3）得到 $\gcd(m,n)$ 的计算量是 $O(\log_2^3 m)$。□

熟知，由（3）式的最后一个等式往回推演，可以得到 u 和 v 使 $\gcd(m,n)=um+vn$，而且计算出 u,v 的计算量也可以证明是 $O(\log_2^3 m)$，故对一次不定方程 $ax+by=c$（其中 $\gcd(a,b)\mid c$）和一次同余式 $ax\equiv c(\mathrm{mod}\, b)$（$\gcd(a,b)\mid c$）求解的计算量是 $O(\log_2^3 m)$，这里 $m=\max(a,b,c)$。

另外，为了以后的需要及其本身的重要性，我们要讨论雅可比符号 $\left(\frac{n}{m}\right)$ 的计算。由定义知 $\left(\frac{n}{m}\right)=\pm 1$，但要确定它是 $+1$ 还是 -1，不是可以显然地得出的，故仍需要一个算法。我们可以利用雅可比符号的两点性质：

i）若 $l\equiv n(\mathrm{mod}\, m)$，则 $\left(\frac{l}{m}\right)=\left(\frac{n}{m}\right)$。

ii）互倒律：$\left(\frac{-1}{m}\right)=(-1)^{\frac{m-1}{4}}$，$\left(\frac{2}{m}\right)=(-1)^{\frac{m^2-1}{8}}$ 及 $\left(\frac{n}{m}\right)=(-1)^{\frac{(m-1)(n-1)}{4}}\cdot\left(\frac{m}{n}\right)$。

写出一个类似于欧几里得算法的算法如下。

以下用 $e(x)$ 表示自然数 x 的最高 2 因子的幂次，即 $2^{e(x)}\mid x$，$2^{e(x)+1}\nmid x$，用 x' 表示 x 的最大奇因

子，显然有 $x=2^{e(x)}\cdot x'$。现在，给定两个数m，n，这里m为奇数且设 $m>n$。令 $r_0=m$，$r_1=n$，则有

$$\left(\frac{r_1}{r_0}\right)=(-1)^{\frac{r_0^2-1}{8}\cdot e(r_1)}\cdot\left(\frac{r_1'}{r_0}\right)$$

$$=(-1)^{\frac{r_0^2-1}{8}\cdot e(r_1)}\cdot(-1)^{\frac{(r_0-1)(r_1'-1)}{4}}\cdot\left(\frac{r_0}{r_1'}\right)$$

$$=(-1)^{\frac{r_0^2-1}{8}\cdot e(r_1)}\cdot(-1)^{\frac{(r_0-1)(r_1'-1)}{4}}\cdot\left(\frac{r_2}{r_1'}\right). \tag{4}$$

其中 $r_2=r_0 \bmod r'_1$，即 r_0 除 r'_1 得的余数。再将 r_2 代换 r_1，r'_1 代换 r_0 重复（4）的手续，不断地重复（4）的演算，最后得到 $r_2=1$ 或 2 为止。这时可用互倒律的前两式得出 $\left(\frac{n}{m}\right)$。不难看出，计算 $\left(\frac{n}{m}\right)(m>n)$ 的计算量是 $O(\log_2{}^3m)$。

（6） 中国剩余定理

定理1.11 设 $m,\cdots,m_r(m_i>1)$ 是两两互素的自然数，$a_1,\cdots,a_r$ 是r个整数，$M=m_1\cdots m_r$，则存在唯一的 $0\leqslant a<M$ 使得 $a\equiv a_i(\bmod m_i)$ $i=1,\cdots,r$。且有算法使得其计算 a 的计算量是 $O(r\cdot\log_2{}^3M)$。

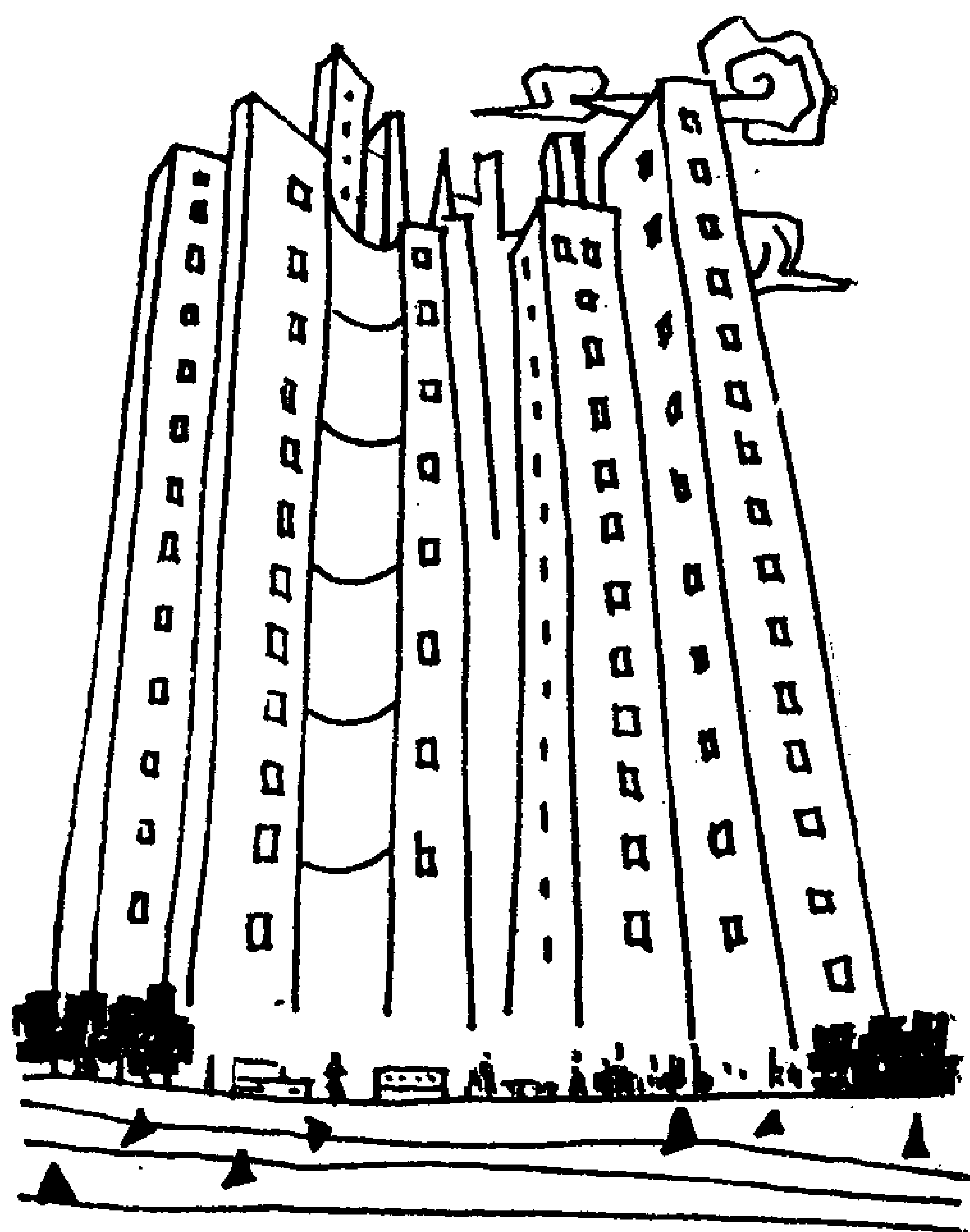
证明 设 $M_i=\frac{M}{m_i}$ $i=1,\cdots,r$，又设 u_i 是 $M_ix\equiv1(\bmod m_i)$ 的解 $(i=1,\cdots,r)$，则 $a=\sum_{i=1}^{r}$

$u_iM_ia_i \bmod M$就满足$0\leqslant a<M$和$a\equiv a_i(\bmod m_i)$，$i=1,\cdots,r$；设a，b都满足$0\leqslant a<M$，$0\leqslant b<M$，$a\equiv a_i(\bmod m_i)$，$b\equiv a_i(\bmod m_i)$ $(i=1,\cdots,r)$，即$a-b\equiv 0(\bmod m_i)$，$i=1,\cdots,r$，而$m_1,\cdots,m_r$两两互素，因而$a-b\equiv 0\ (\bmod M)$，再由$0\leqslant a<M$，$0\leqslant b<M$即得$a=b$。于是唯一性证得。

按上面的方法计算a，先要计算$M=m_1\cdots m_r$，因为$m_1\cdots m_i$乘m_{i+1}的计算量是$O(\log_2 m_1\cdots m_i\cdot\log_2 m_{i+1})$，故计算$M$的计算量是$O(\sum_{i=1}^{r-1}\log_2 m_1\cdots m_i\cdot\log_2 m_{i+1})\leqslant O(\sum_{i=1}^{r-1}\log_2 M\cdot\log_2 m_{i+1})(=O(\log_2^2 M)$。同样，计算$M_i$的计算量也不多于$O(\log_2^2 M)$。由上一段的讨论，求得$u_i$的计算量是$O(\log_2^3 M_i)\leqslant O(\log_2^3 M)$。最后，由$u_i$，$M_i$，$a_i$计算$a$的计算量也是$O(\log_2^3 M)$，因而求$a$的总计算量是$O(r\log^3_2 M)$。 □

关于数论中的基本算法的研究，是计算数论的最基本的研究，不仅是它本身很重要，它在很多其它分支如计算机科学、代数学等中有很大的用处。然而到目前为止，除了有些文章散见于某些杂志和书籍中，还没有较完全的书来专门讨论这些问题。在这一方面，最好的文章是勒默的"计算机技巧应用于数论"，见[3]。

二　素性判别



素数这个概念，早在公元前很多世纪就为人们所熟知。后来人们发现所有自然数都是由素数乘起来得到的。欧几里得证明了素数有无限多个，因此，任意大的素数都存在。可是，在自然数的序列1，2，3，……中，素数和合数混杂在一起，对前数千个素数的分布之考察发现素数的分布没有规则，因此，鉴别一个自然数是素数还是合数就成为问题。这个问题在中世纪就引起人们的注意。当时人们试图寻找素数公式。到高斯时代，基本上确认了简单的素数公式是不存在的。在那时，即使对一个十位数的整数来作素性判别都是相当困难的，因此，高斯认定素性判别是数论中最困难的问题之一。从此后，这个问题吸引了大批数学家，但当时的人们没有计算机这个有力的工具，对一般十位以上的数都束手无策。因此，他们或者只对很特殊的数作了些研究，或者，对素性判别作了一般性讨论。而真正用得出的结论去判别一个大数是否素数时，常常因为计算量太大而归于失败。到本世纪初，手摇计算机

的产生帮助了象勒默等人发展素性判别。而在1950年之后，由于电子计算机的诞生，数学家们又将注意力转到素性判别的问题上来了。目前，这个分支已经产生了很多好结果。我们将在此作介绍。因为素性判别的理论真正得到发展是近几十年的事，因此，我们主要是介绍近几十年的工作。在本书中，素性判别和素数判别指的是同一件事。

1．素性判别的一般理论

素性判别的算法是指一个算法，用它可以判别任意一个自然数是否素数。迄今为止，素性判别的方法有很多种，但它们有共同的形式，我们试将它们从总体上来讨论。

欲要寻求一个素性判别的算法，应先注意到素数所应该满足的一些性质，即一些必要条件。根据这些性质设计出一个条件组(也称试验组)。这个条件组有两个特点：凡是素数就满足条件组中的每个条件（也称为通过条件组），凡通过这组试验组数，若不是素数，则它必有一个真因子落在某个特定的集合中。现对任给的数n，先看n是否通过试验组，如果不通过，则n是合数；如果通过，则其可能的真因子有一个落在特定的集

合中。然后，用这个特定的集合的每个元素去试除n，若有某个元素不等于1和n且整除n，则n是合数，否则，n是素数。这样的素性判别算法的计算量由两部分构成：对n逐个检验它是否满足条件组中的条件的计算量和对特定的集合中的元素逐个试除n。设第一部分的计算量是O（f(n)），特定集合的元素个数是g（n），（一般这g（n）个元素是1到n之间的数）则这个算法的计算量是$O(f(n)+g(n)\log_2^2 n)$。

例：威尔逊证明了n是素数的充要条件是$(n-1)!+1\equiv 0\ (\mathrm{mod}\,n)$。于是可设计试验组为"条件$(n-1)!+1\equiv 0\ (\mathrm{mod}\,n)$"，而特定集合是$\{1, n\}$。若n通过试验组，则n的可能的真因子在$\{1, n\}$。但$\{1, n\}$中没有n的真因子，因此，n是素数。检验条件$(n-1)!+1\equiv (\mathrm{mod}\,n)$的计算量是$O(n\log_2^2 n)$。因此，威尔逊的算法的计算量是$O(n\log_2^2 n+2\log_2^2 n)=O(n\log_2 n)$。下面知道，这个算法的计算量太大了，不是有效算法。

到目前为止，仍然没有一个素性判别的多项式算法，换言之，没有一个素性判别的算法，它对n执行时的计算量是$O(P(\log_2 n))$，其中$P(x)$是多项式函数。"是否存在素性判别的多项式算法？"是一个没有解决的公开问题。人们偏向于

说存在素性判别的多项式算法，但至今没有找到。在已有的判别算法中，或者 $f(n)$ 不是 $\log_2 n$ 的多项式，或者 $g(n)$ 不是 $\log_2 n$ 的多项式。因而要得到一个素性判别的多项式算法，就需要设法使上述的 $f(n)$ 和 $g(n)$ 都是 $\log_2 n$ 的多项式。

2. 一个经典的结果

所谓素数，就是除1和其本身之外，不被任何数除尽的数。给定一个数 $n>1$，我们知道凡是大于 n 的数都不能除尽 n，故要检验 n 是否素数，只要用2，3，…，$n-1$ 去试除 n，如果其中有一个整除 n，则 n 是合数，否则，n 是素数。然而，我们可以不要用这么多的数去试除 n。我们先引入下面的结论。

定理 2.1 $n>1$ 是素数当且仅当不大于 $\sqrt{n}$ 的所有素数都不能整除 n。

证明 若 $n>1$ 是素数，则显然不大于 $\sqrt{n}$ 的所有素数不能整除 n，反过来，若 n 不是素数，设 p 是 n 的最小素因子，即 $1<p<n$。这时有 $p\leqslant\sqrt{n}$。否则，若 $p>\sqrt{n}$，因为 $\frac{n}{p}>1$ 是一个自然数，故存在素因子 p_1。显然，p_1 也是 n 的素因子，因此，$p_1\geqslant p>\sqrt{n}$ 且有 $\frac{n}{p}\geqslant p_1$，故

$n \geqslant pp_1 > \sqrt{n} \cdot \sqrt{n} = n$。这是不可能的。因此，若不大于$\sqrt{n}$的所有素数不能整除 n，则 n 是素数。 □

定理 2.1 等价于说，若 $n > 1$ 不是素数，则 n 至少有一个不大于$\sqrt{n}$的因子。因此，对于 $n > 1$，要判别 n 是否素数，只要用不大于$\sqrt{n}$的所有素数去试除 n，若其中有一个素数整除 n，则n是合数，否则 n 是素数。在实际使用时，如果是对不多于 16 位的数 n 来进行素性判别，这时，$\sqrt{n} < 10^8$，故小于$\sqrt{n}$的素数一般都储存在计算机里。但当 n 多于 16 位时（这时$\sqrt{n} > 10^8$），一般计算机没有储存这些小于$\sqrt{n}$的全部素数。因此要先求出小于$\sqrt{n}$的全部素数，再用它们去试除 n。但为了避免分两步进行，有时，就干脆将小于$\sqrt{n}$的全部奇数去试除 n 来判别 n 是否素数。

以上描述的算法是最直接的、最简单的素数判别法——试除法。用第一节的语言来说，试除法所依赖的试验组就只有"条件 $n > 1$"；而其特定的集合是不大于待判别数的平方根的所有自然数。注意到："$n > 1$"这个条件的验证是可以不考虑计算量的，故得到试除法对 n 执行的计算的计算量是$O(\sqrt{n}\log_2^2 n)$。由此可见，它比威尔逊的方法好些。

试除法是一个相当古老的判别法。与它相关的一个寻找素数的方法是埃那托森斯筛法，这个古老的筛法在造素数表时，仍然起着很大的作用。给定一个自然数 n，要找出不大于 n 的所有素数，可以用如下的埃那托森斯筛法：将1，2，…，n 按自然顺序排列好。第一步："删去第一个未被删去或圈住的数。"第二步："将第一个未被圈住和删去的数圈住，删去所有这个刚被圈住的数的倍数。"在执行第一次第一步时，是删去1，第一次执行第二步时，圈住 2 并删去 2 的倍数。然后回转重复第二步、第二步、……。这样若干次执行第二步直到不大于$\sqrt{n}$的每个数都被删去或圈住为止，这时，被圈住的和剩下来未被删去和圈住的数便是不大于 n 的全部素数。下面的例子说明了本节的内容的应用。

例 判别7393是否素数。

$[\sqrt{7393}]=85$，先用埃那托森斯筛法找出不大于85的所有素数：将 1 至85的数按自然顺序排列好，然后，循环地执行上述的第二步直至不大于$\sqrt{85}\approx 9$的数全被删去或圈住为止。

（下面划了两、三道斜线的数分别被两、三次删去），即得85之下的素数是 2 、3 、5 、 7 、11、13、17、19、23、29、31、37、41、43、47、53、59、61、67、71、73、79、83。

~~1~~ ② ③ ~~4~~ ⑤ ~~6~~ ⑦ ~~8~~

~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~

19 ~~20~~ ~~21~~ ~~22~~ 23 ~~24~~ ~~25~~ ~~26~~ ~~27~~ ~~28~~

29 ~~30~~ 31 ~~32~~ ~~33~~ ~~34~~ ~~35~~ ~~36~~ 37 ~~38~~

~~39~~ ~~40~~ 41 ~~42~~ 43 ~~44~~ ~~45~~ ~~46~~ 47 ~~48~~

~~49~~ ~~50~~ ~~51~~ ~~52~~ 53 ~~54~~ ~~55~~ ~~56~~ ~~57~~ ~~58~~

59 ~~60~~ 61 ~~62~~ ~~63~~ ~~64~~ ~~65~~ ~~66~~ 67 ~~68~~

~~69~~ ~~70~~ 71 ~~72~~ 73 ~~74~~ ~~75~~ ~~76~~ ~~77~~ ~~78~~

79 ~~80~~ ~~81~~ 82 83 ~~84~~ ~~85~~

用上列素数去试除7393，经过验算，其中没有一个能整除7393，因此，根据试除法，7393是一个素数。

埃那托森斯筛法用于造小于n的素数表时，执行第二步的次数为$\frac{2\sqrt{n}}{\log_2 n}$，但是，因为要将1，2，…，n全部排列下来，因此，它对计算机的容量要求是至少n个空间，当n很大时，（例$n=10^{15}$）一般计算机都难提供这么大的空间。因

此，目前制出的素数表还仅在8位数以下。

3. 费马小定理和卡米歇尔数

试除法出现之后，一直到16世纪，其间除了一些很特殊的、很局限的素性判别法外，没有什么重要的结果。但到1640年，法国数学家费马首先注意到素数的一个性质，那就是下面讲叙的费马小定理。这个性质是以后的所有素性判别法产生的根源。

定理 2.2 （费马小定理） 若n是素数，则对所有不被n整除的a，有 $a^{n-1} \equiv 1 \pmod{n}$。

证明 因为 $(a,n)=1$，则 $a, 2a, \cdots, (n-1)a$ 分别按某个重排顺序模n同余于1，2，…，n－1。故有 $a^{n-1} \cdot (n-1)! \equiv (n-1)! \pmod{n}$。因为n是素数，因此n与 $(n-1)!$ 互素，因此，$a^{n-1} \equiv 1 \pmod{n}$。 □

我们将费马小定理的另一个形式写成下面的推论。

推论 若n是素数，则对任意的整数有 $a^n \equiv a \pmod{n}$。

由第一章第二节的讨论，我们知道，对某个自然数a，$1 \leqslant a < n$，要验证 $(a,n)=1$ 和 $a^{n-1} \equiv 1 \pmod{n}$ 是否成立，需要计算量为 $O(\log_2^3 n)$。

因此，以对某些 a，$1 \leqslant a < n$，$(a, n) = 1$ 验证 $a^{n-1} \equiv 1 \pmod{n}$ 作为试验组，将可能产生较好的素数判别法。为此，我们要来看 n 通过试验组的话，n 具备什么性质。这就要考察一下费马小定理的逆命题。

大约2500年前，中国古代数学家就发现 2^2-2 是2的倍数，2^3-2 是3的倍数，2^5-2 是5的倍数，2^7-2 是7的倍数，$2^{11}-2$ 是11的倍数，而2，3，5，7，11都是素数。故他们由此肯定："若 2^n-2 是 n 的倍数（即 $2^n \equiv 2 \pmod{n}$），则 n 是素数"。莱布尼兹研究了《易经》中的这一记载之后，也相信了这个结果。假如这个结果是正确的，则对任给的自然数 $n > 1$，只要验证 $2^n \equiv 2 \pmod{n}$ 是否成立，即可判定 n 是否素数，其计算量是 $O(\log_2^3 n)$。这就有了素性判别的多项式算法。然而，很不幸运，这个结果不是正确的。1819年，法国数学家赛路斯指出，$2^{341} \equiv 2 \pmod{341}$，但是 $341 = 11 \times 31$ 是个合数。自此以后，人们发现了许多具有不同底值 a 的反例，如 $3^{91} \equiv 3 \pmod{91}$ 但 $91 = 7 \times 13$，$4^{15} \equiv 4 \pmod{15}$ 但 $15 = 3 \times 5$ 等等。事实上，对于任意的 a，都有这样的反例，而且有无限多个。我们将证明这一点，在此之前，先给出一个定义。

定义 对整数 $a>1$，满足 $a^n \equiv a \pmod{n}$ 的合数 n 称为底为 a 的伪素数。

定理 2.3 对每一个整数 $a>1$，有无限多个底为 a 的伪素数。

证明 给定 $a>1$，设奇素数 $p \nmid a(a^2-1)$，令 $n=\frac{a^{2p}-1}{a^2-1}$。则 n 是自然数，且 $n=\frac{a^p-1}{a-1}\cdot\frac{a^p+1}{a+1}$，故 n 是合数。下面，我们要来证明 n 是底为 a 的伪素数。我们有 $(a^2-1)(n-1)=a^{2p}-a^2=a\cdot(a^{p-1}-1)(a^p+a)$。由于 a^p 与 a 的奇偶性相同，故 $2 \mid a^p+a$；又由费马小定理和 $p \nmid a$ 得 $p \mid a^{p-1}-1$；由于 p 是奇数，则 a^2-1 整除 $a^{p-1}-1$；而 $p \nmid a^2-1$，故有 $p(a^2-1) \mid a^{p-1}-1$。因此，$2p(a^2-1) \mid (a^2-1)(n-1)$ 即 $2p \mid n-1$。令 $n=1+2pm$。现在有 $a^{2p}=n\cdot(a^2-1)+1\equiv 1 \pmod{n}$。故 $a^{n-1}=a^{2pm}\equiv 1^m=1 \pmod{n}$。即 $a^n\equiv a \pmod{n}$，因此 n 是底为 a 的伪素数。由于对每个 $a>1$，满足 $p \nmid a(a^2-1)$ 的奇素数 p 有无限多个，故以上作出的 $n=\frac{a^{2p}-1}{a^2-1}$ 也有无限多个。因此，以 a 为底的伪素数有无限多个。

虽然底为 2 的伪素数有无限多个，但是，它们相对于素数而言是相当少的，隆德大学的波赫

曼证明了小于$10^{10}\times 2$的素数有882206716个，而塞尔弗来季和瓦格斯塔夫计算出底为2的伪素数在1到2×10^{10}之间只有19865个。因此，在大多数情况下，我们的断言“如果$2^n\equiv 2\pmod{n}$，则n是素数。”是正确的，用这个断言作素性判别，出错的概率很小，例如在$n<2\times 10^{10}$的范围内，出错的概率小于$19865/(882206716+19865)\approx 0.0000025$。不过，这样的结果，在实际用来作素性判别时，用处不大，尽管它出错的概率很小，但它毕竟是会出错的。而且，对于特定的一个n，用上面的断言去判别它的素性，就不能排除错误就在此出现。（对于其它底a，可以作同样讨论。）

令人惊奇的是，存在这样的合数n，对任意的满足$(a,n)=1$的$a>1$，n都是底为a的伪素数。这样的合数的存在是费马小定理的逆命题不成立的最合适的例证。因为它说明，即使对所有满足$(a,n)=1$的a有$a^{n-1}\equiv 1\pmod{n}$，仍不能断定n是素数。这样的合数是由卡米歇尔首先发现的，故叫卡米歇尔数。

例 561是卡米歇尔数。这是卡米歇尔发现的第一个卡米歇尔数，也是最小的卡米歇尔数。因为$561=3\times 11\times 17$，若$\gcd(a,561)=1$，则$\gcd(a,3)=\gcd(a,11)=\gcd(a,17)=1$，

由费马小定理知，$a^2 \equiv 1 \pmod 3$，$a^{10} \equiv 1 \pmod{11}$，$a^{16} \equiv 1 \pmod{17}$。由于$\mathrm{Lcm}(2, 10, 16) = 80$，故$a^{80} \equiv 1 \pmod 3$，$a^{80} \equiv 1 \pmod{11}$，$a^{80} \equiv 1 \pmod{17}$，即得$a^{80} \equiv 1 \pmod{561}$，故$a^{560} \equiv (a^{80})^7 \equiv 1^7 = 1 \pmod{561}$。所以561是卡米歇尔数。

一般地，我们有下面的定理。

定理 2.4 （卡米歇尔） n是卡米歇尔数的充分必要条件是：i) n无平方因子；ii) n的每一个素因子p有$p-1 \mid n-1$；iii) n是奇数且至少有三个不同的素因子。

证明 先证明充分性。设n满足条件i), ii), iii)，令$n = p_1 p_2 \cdots p_k$ $(k \geq 3)$，$p_1, p_2, \cdots, p_k$是互不相同的奇素数。现对任意的$a > 1$，若$(a, n) = 1$，则$(a, p_i) = 1$，由费马小定理得$a^{p_i - 1} \equiv 1 \pmod{p_i}$，$i = 1, 2, \cdots, k$，而由条件ii)，有$\mathrm{Lcm}(p_1 - 1, p_2 - 1, \cdots, p_k - 1) \mid n - 1$，故得$a^{n-1} \equiv 1 \pmod{p_i}$ $(i = 1, 2, \cdots, k)$，因而$a^{n-1} \equiv 1 \pmod n$。故n是卡米歇尔数。

必要性的证明：设n是卡米歇尔数。令$n = p_1^{u_1} \cdots p_k^{u_k}$，其中$u_i \geq 1$，$i = 1, \cdots, k$。先证明n是奇数。若n是偶数且含有一个奇素因子p。这时，取a是p的奇原根，由$a^{n-1} \equiv 1 \pmod n$得$a^{n-1} \equiv 1 \pmod p$。因此，$p-1 \mid n-1$，但$p-1$为偶

数而$n-1$为奇数，这是不可能的。又若$n=2^t$，$t>1$，取$a=3$，则$(3,n)=1$。但$3^{2^{t-1}}\not\equiv 1 \pmod{2^t}$。否则，若$3^{2^{t-1}}\equiv 1 \pmod{2^t}$，则$3^{2^t}\equiv 3 \pmod{2^t}$，但$3^{2^t}=(2+1)^{2^t}=\sum_{i=0}^{2^t}\binom{2^t}{i}2^i\equiv 1 \pmod{2^t}$，故得$3\equiv 1 \pmod{2^t}$，即得$t=1$，这与假设不符。故$n$必是奇数，因而$p_i$是奇素数，$i=1,\cdots,k$。令$g_i$是模$p_i^{u_i}$的原根，$i=1,\cdots,k$。由中国剩余定理，可求得$a$使$a\equiv g_i \pmod{p_i^{u_i}}$ $(i=1,\cdots,k)$。故$\gcd(a,n)=1$。因为n是卡米歇尔数，则$a^{n-1}\equiv 1 \pmod n$，即有，$a^{n-1}\equiv 1 \pmod{p_i^{u_i}}$，另一方面$a^{n-1}\equiv g_i^{n-1} \pmod{p_i^{u_i}}$，故有$g_i^{n-1}\equiv 1 \pmod{p_i^{u_i}}$。由原根的定义得$\varnothing(p_i^{u_i})\mid n-1$，即是$p_i^{u_i-1}(p_i-1)\mid n-1$，$i=1,\cdots,k$。而$n-1=p_1^{u_1}\cdots p_k^{u_k}-1$，故$u_i=1, i=1,\cdots,k$，因此，条件i)证得。又$p_i-1\mid n-1, i=1,\cdots,k$，即条件ii)证得。对iii)，若$k<3$，由$n$是合数，则$k=2$，则有$n=p_1\cdot p_2$ $(p_1\neq p_2)$，由于$p_1-1\mid p_1p_2-1$，而$p_1p_2-1=(p_1-1)p_2+p_2-1$，故得$p_1-1\mid p_2-1$，同理可得，$p_2-1\mid p_1-1$，由此可得$p_1-1=p_2-1$，即$p_1=p_2$，这与假设不符，因而$k\geq 3$，即iii)证得。 □

例 由定理2.4可得$2821=7\cdot 13\cdot 31$，$10585=5\cdot 29\cdot 73$，$27845=5\cdot 17\cdot 29\cdot 23$，$172081=7\cdot 13\cdot 31\cdot$

61都是卡米歇尔数。

由于发现了卡米歇尔数，人们认识到利用费马小定理来作素性判别，不是件简单的事情。假如卡米歇尔数只有有限个，则可以给出一个上界M，这样，在n＞M的范围内对n作素性判别就变得容易起来。然而，卡米歇尔数可能有无限多个，这只是一个猜想，至今无人给出证明，但人们大都认为这个结果是正确的。

4. 从努卡斯到威廉斯

费马小定理的逆命题不成立，使得人们利用它来作素性判别出现困难，但是，因为它所提供的检验组中的条件是很容易验证的，数学家们并没有放弃利用这个优美的定理来产生素性判别的方法。终于，到1876年，法国数学家努卡斯在增加条件的情况下，给出了费马小定理的一种形式的逆命题。他证明了下面的结果。

定理 2.5 （努卡斯） 对自然数n，设$n-1=q_1^{\alpha_1}\cdots q_t^{\alpha_t}$，若有a使得

(L) $$a^{n-1}\equiv 1\ (\mathrm{mod}\, n)\quad 而\ a^{\frac{n-1}{q_i}}\not\equiv 1\ (\mathrm{mod}\, n),$$

$i=1,\cdots,t$。则n是素数。

证明 设a对模n的次数是e。由欧拉定理得$e\mid\varphi(n)$。由定理叙述中的条件(L)得$e\mid n-1$，

但$e \nmid \frac{n-1}{q_i}$, $i=1,\cdots,t$, 因此, $q_i^{u_i} | e, i=1,\cdots,t$, 即得$n-1 | e$。故有$n-1 | \varnothing(n)$, 这就说明n是素数。 □

给定一个自然数n, 如果恰巧 $n-1$可以很容易地被完全分解, 则可用定理2.5来判别n是否素数。这时, 只需要去寻找一个a使其满足条件(L)。若不存在这样的a, 则n就不是素数。〔因为素数的原根就满足条件(L)〕; 若有这样的a存在, 则有$\varnothing(n-1)$个。困难在于如何选取合适的a来验证条件(L)。在实践中, 常常是随机地选取a来试验, 直到有某个a满足条件为止。即产生了一个素性判别的概率算法。

例 判别4093是否素数。

取$a=2$。验证条件(L): $2^{4092} \equiv 1 \pmod{4093}$,

$2^{\frac{4092}{2}} \equiv -1 \pmod{4093}$, $2^{\frac{4092}{3}} \equiv -360 \pmod{4093}$,

$2^{\frac{4092}{11}} \equiv 3024 \pmod{4093}$, $2^{\frac{4092}{31}} \equiv 1121 \pmod{4093}$,

这里 $4092 = 4093 - 1 = 2^2 \cdot 3 \cdot 11 \cdot 31$。由定理2.5得, 4093是素数。

后来, 塞尔弗来季对努卡斯的结果稍加推广, 得到结果: "设$n-1 = q_1^{u_1} \cdots q_t^{u_t}$, 若对每个$q_i$, 存在$a_i$使$a_i^{n-1} \equiv 1 \pmod{n}$, $a_i^{\frac{n-1}{q_i}} \not\equiv 1 \pmod{n}$, 则n是素数。" 这里所要求的条件比定理2.5中的

条件(L)容易被满足些。

我们注意到，努卡斯的结果及塞尔弗来季的推广都要求待判别的数减去1后可以容易地完全分解。可是，对于很大的数来说，这一点往往办不到。然而，在很多情况下，它可以部分地分解。这时，可否利用已分解了的部分来作素性判别呢？在这一方面首先出现的是普罗丝的一个结果。

定理 2.6 （普罗丝） 设 n 是奇数，若 $n-1=mq$，其中q是一个奇素数且满足 $2q+1>\sqrt{n}$，又若有a使 $a^{n-1}\equiv 1\pmod{n}$，$a^m\not\equiv 1\pmod{n}$，则n是素数。

证明 因为$a^{n-1}\equiv 1\pmod{n}$，$a^m\not\equiv 1\pmod{n}$，则同定理2.5的证明可得 $q\mid\varnothing(n)$。设n的标准分解式是 $n=\prod_{i=1}^{s}n_i^{\alpha_i}$，则$\varnothing(n)\mid n\cdot\prod_{i=1}^{s}(n_i-1)$。

故$q\mid(mq+1)\cdot\prod_{i=1}^{s}(n_i-1)$。即有某个$n_i$使$q\mid n_i-1$。又$2\mid n_i-1$，故有$2q\mid n_i-1$，即$n_i\equiv 1\pmod{2q}$。由于$n\equiv 1\pmod{2q}$，则有$\frac{n}{n_i}\equiv 1\pmod{2q}$。若$n\neq n_i$，则$\frac{n}{n_i}\neq 1$，故$\frac{n}{n_i}>1+2q$，

因此 $n=n_i\cdot\frac{n}{n_i}>(2q+1)\cdot(2q+1)>\sqrt{n}\cdot\sqrt{n}$

$=n$。矛盾。这说明$n=n_i$，即n是素数。

例 判别823001是否素数。

令$n=823001$，则$n-1=823000=1000\times 832$。令$m=1000$，$q=823$，则$2q+1>\sqrt{n}$，且$2^{n-1}\equiv 1 \pmod n$，而$2^{1000}\not\equiv 1 \pmod{823001}$，故由普罗丝的结果得，823001是素数。

1914年，波克林顿进一步得到了下面的更好的结果。

定理 2.7 （波克林顿） 对自然数n，设$n-1=F_1R_1$，其中F_1是$n-1$的已经分解出的部分（即已知的素因子的适当幂次的乘积），R_1是$n-1$的未分解出的部分（即$\frac{n-1}{F_1}=R_1$是一个数，我们不知它是素数还是合数，或者知道它是合数，但没分解出任何因子来）$(F_1, R_1)=1$。若对F_1的每个素因子q_i，存在a_i使得

$$(P) \qquad a_i^{n-1}\equiv 1 \pmod n, \quad (a_i^{\frac{n-1}{q_i}}-1, n)=1$$

则n的每个素因子p都满足$p\equiv 1 \pmod{F_1}$。

证明 设p是n的素因子，e_i是a_i对模p的次数。则$e_i \mid p-1$，而因为$a_i^{n-1}\equiv 1 \pmod p$，则$e_i \mid n-1$。由$(a_i^{n-1/q_i}-1, n)=1$，故$a_i^{n-1/q_i}\not\equiv 1 \pmod p$，故$e_i \nmid \frac{n-1}{q_i}$，因此$q_i^{\alpha_i} \mid e_i$，其中$q_i^{\alpha_i} \parallel F_1$。以上证明了，对每个整除$F_1$的素因子$q_i$有

$q_i^{\alpha_i} \mid p-1$且$q_i^{\alpha_i} \parallel F_1$。即得$F_1 \mid p-1$。即 $p \equiv 1 \pmod{F_1}$。 □

推论 若奇数 n 满足定理2.7的条件且 $F_1 > \sqrt{n}$，则n 是素数。

由定理 2.7，读者不难推出这个推论。波克林顿的结果及其推论在判别素数时很有用。但是，通常"$F_1 > \sqrt{n}$"这个条件因为 $n-1$ 的分解不够而不能满足，而进一步分解 $n-1$ 又有困难，这时要判别n是素数否用定理2.7 及其推论还不够。因而，人们就进一步改进波克林顿的结果。到1978年，勒默、卜利尔哈特等人得到下面的更好的结果。

定理 2.8 设n 是奇数且满足定理 2.7 的条件，$m \geqslant 1$，当 $m > 1$ 时还假定 $\lambda F_1 + 1 \nmid n$，$\lambda = 1, \cdots, m-1$，如果

(B) $\quad n < (mF_1 + 1)[2F_1^2 + (r-m)F_1 + 1]$

此处 r 和 s 由 R_1 和F_1 如下确定：$R_1 = 2F_1 s + r$，$1 \leqslant r < 2F_1$（即 r和s 分别是用$2F_1$ 除 R_1 所得的余数和商），则 n 是素数的充要条件是 $s = 0$ 或 $r^2 - 8s \neq □$，□表示平方数。

证明 我们将证明结论的等价形式：n 是合数的充要条件是 $s \neq 0$ 且$r^2 - 8s = □$。

（i）（$\Longrightarrow$）由定理2.7知，n 的每个因子模F_1同余于1。若n是合数，则可写 $n = (cF_1 + 1)$

(dF_1+1)，$c, d\geqslant m$。注意到 R_1 是奇数，F_1 是偶数，则由 $R_1=\frac{n-1}{F_1}=cdF_1+c+d$ 得，$c+d$ 是奇数，cd 是偶数。而由

$$cdF_1+c+d=R_1=2F_1s+r \qquad (*)$$

得

$$c+d\equiv r(\bmod 2F_1) \qquad (**)$$

因为 r 是 $2F_1$ 除 R_1 的最小非负剩余，因此，$c+d-r\geqslant 0$。另一方面，由 $(c-m)(d-m)\geqslant 0$ 得 $cd\geqslant m(c+d)-m^2$，进而有

$$\begin{aligned}(mF_1+1)(2F_1^2+(r-m)F_1+1)&>\\ n=cdF_1^2+(c+d)F_1+1&\\ \geqslant(m(c+d)-m^2)F_1^2+(c+d)F_1+1&\\ =(mF_1+1)((c+d-m)F_1+1)&\end{aligned}$$

即得 $2F_1^2+(r-m)F_1+1>(c+d-m)F_1+1$，即 $c+d-r<2F_1$ 故由 $(**)$ 得 $c+d=r$，再由 $(*)$ 得 $2s=cd\neq 0$，且

$$r^2-8s=(c+d)^2-4cd=(c-d)^2=\square$$

(ii) $(\Longleftarrow)$ 当 $s\neq 0$，$r^2-8s=t^2$ 时，$n=F_1R_1+1=F_1(2sF_1+r)+1=[(r^2-t^2)F_1^2/4]+rF_1+1=(\frac{r-t}{2}F_1+1)(\frac{r+t}{2}F_1+1)$，故 n 是合数。

现在举一个例子来说明定理 2.8 的用途。历

史上，在1904年出版的坎特伯雷难题集上，列了一个难题是：$\overbrace{111\cdots1}^{19个}$是否素数？这个难题当时无人可以解答，现在，我们用定理 2.8 就可以解答它了。设 $n=\overbrace{111\cdots1}^{19个}$，则$n-1=F_1R_1$，其中$F_1=3333330=2\times3^2\times5\times11\times37\times91$。$R_1=2F_1s+r$，其中 $s=500000, r=3666667, r<2F_1$。并且 $r^2-8s=13444450488889$，它被 7 整除但被49除余14，即不是完全平方数。取 $m=1$，这时有

$$\begin{aligned}&(mF_1+1)(2F_1^2+(r-m)F_1+1)\\&=3333331\times(2\times3333330^2+3666666\\&\times3333330+1)\\&=3333331\times(10333326\times3333330+1)\\&=3333331\times34444385555581>n\end{aligned}$$

而且$3^{n-1}\equiv1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{2}}\not\equiv1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{3}}\neq1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{5}}\not\equiv1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{11}}\not\equiv1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{37}}\not\equiv1(\mathrm{mod}\,n)$，$3^{\frac{n-1}{91}}\not\equiv1(\mathrm{mod}\,n)$。由定理2.8知，n是素数。

有时，对某些大数n 作素性判别时，F_1还可能不够大，不足以使不等式（B）得以满足，这时，还可以利用R_1的可能的素因子的下界来帮助我们对 n 作素性判别。下面我们给出勒默和卜利尔哈特等人的一个结果，其证明与定理 2.8 的证明类似，只是稍繁一些，故略去不写。

定理 2.9 （勒默、卜利尔哈特） 对自然数n设$n-1=F_1R_1$，其中F_1,R_1同定理2.7之意。设n满足定理2.7的条件且存在a使$a^{n-1}\equiv1(\mathrm{mod}\,n)$但是$(a^{\frac{n-1}{R_1}}-1,n)=1$，设$R_1$的素因子至少大于$B_1$。若不等式

$$(B')\qquad n<(B_1F_1+1)[2F_1^2+(r-B_1)F_1+1]$$

满足，其中r,s同定理2.8中的r,s。则n是素数当且仅当$s=0$或$r^2-8s\neq\square$。

努卡斯在给出定理2.5所叙述的结果的同时，他注意到幂数列$\{a^n\}=\{a^0,a^1,a^2,\cdots\}$可以看作他所定义的努卡斯序列的特殊情况（$P=a,Q=0$），而且还注意到幂的计算和努卡斯序列的项的计算很相似（如第一章第二节所述），特别是他发现了努卡斯序列的一个类似于费马小定理的性质，即是："设$\{U_n\}$是由P，Q决定的努卡斯序列，$p\nmid 2Q$，p是一个素数，则$U_{p-\varepsilon_p}\equiv0(\mathrm{mod}\,p)$，其中$\varepsilon_p=\left(\frac{D}{p}\right)$是勒让德符号，$D=P^2-4Q$。"其证明见[1]。努卡斯对这个性质作了不少研究，得到不少好结果，其中就有下面的一个结果。

定理2.10 对自然数n，设$n+1=\prod_{i=1}^{s}q_i^{\beta_i}$，若对每个$q_i$存在一个努卡斯序列$\{U_n^{(i)}\}$（但它们的判别式$D=P_i-4Q_i$是固定的，即不依赖于i）使

得 $\left(\frac{D}{n}\right)=-1$，且$n \mid U_n^{(1)}$而$n \nmid U_{\frac{n+1}{q_i}}^{(1)}$，则 n 是素数。

我们将定理2.5和定理2.10相比，发现两者形式上很相似，区别是前者是用$n-1$的分解式和一组幂运算的检验组，而后者是用$n+1$的分解式和一组努卡斯序列的项运算的检验组．并且努卡斯的定理2.10之证明也是从定理2.5的证明中得到启发的，他利用努卡斯序列的一些与幂模相类似的性质及一个与模次数相当的概念，所谓“幻秩”来证明定理 2.10。努卡斯序列和幻秩的讨论已超出本书范围，而且很繁锁冗长，故不予在此讨论，定理2.10 的证明也免于写出．有兴趣的读者可参见威廉斯的文章[4]．不过，至此我们已经感觉到，可以沿着从定理2.5到定理 2.9 的同样的思路去进一步讨论．即：利用$n+1$的部分因子，用努卡斯序列代替幂运算，可以得到与普罗丝、波克林顿的结果相对应的结果（这样的结果于1975年由莫利桑得到了）．用 $n+1$ 的部分因子和用努卡斯序列代替幂运算作为检验条件，可以得到与定理 2.8 和定理 2.9 相当的结果．这是勒默、卜利尔哈特等人的工作．更有意思的是他们将这两种结果结合在一起，也就是说，同时利用 $n-1$ 和 $n+1$ 的因子分解，同时用幂和努卡斯序

列两种方式的检验条件，来作对 n 的素性检验。

继勒默和卜利尔哈特等人之后，威廉斯和加得等人注意到，有一部分自然数n，其$n\pm1$很不容易被分解，但是n^2+1或 $n^2\pm n+1$ 的因子却容易分解出来。或者，对某些n，虽然 $n\pm1$可以分解一些因子出来，但所分解出来的因子还不足以满足用上面讲述的方法作 n 的素性判别的条件，这时，就要设法利用n^2+1或$n^2\pm n+1$ 的因子来判别 n 的素性。威廉斯和加得等人对勒默和卜利尔哈特的工作作了仔细的研究后，发现可以利用一种推广的努卡斯序列（勒默对努卡斯序列的推广工作）来建立用 n^2+1，$n^2\pm n+1$ 的因子作素性判别的方法。他们得到了很多结果。这些结果的形式同定理2.5到定理2.9 相类似，只是将 $n-1$ 相应地换成n^2+1或$n^2\pm n+1$，将诸如（P），(B)，（B′）等检验条件组相应地换成用推广的努卡斯序列表述的条件组。遗憾的是，即使不加证明地叙述出这些结果也要占去大量篇幅，故此处略去不讲。有兴趣者可参见〔2〕。

从努卡斯到威廉斯的工作，是一条人们探索素性判别方法的漫长之路。它从努卡斯开始，经过许多数学家的努力，最后由威廉斯宣告此路已走到头了。威廉斯于 1978 年在文章〔2〕中说道：“我们认为，沿这个方向的工作，我们已经

作了尽可能的推广，要取得素性判别的更进一步的结果，或许应该朝其他方向上努力。”也就是说，威廉斯认为，要建立用n的其他整系数多项式f(n)（别于$n\pm1, n^2+1, n^2\pm n+1$）的因子分解来作n的素性判别的方法是不可能的。

从努卡斯到威廉斯的一系列工作，用于对少于70位的自然数作素性判别是很有效的，有时甚至比后面介绍的艾德利曼等人的方法快得多，故它们有不少用处。但是，对于70位以上的数作素性判别，它们就显然无能为力了。因而，如威廉斯本人所说，数学家们应该进一步研究新的素性判别法。

5. 素性判别与广义黎曼猜想

1976年，缪内发现了素性判别与广义黎曼猜想（见附录）的一个深刻的关系。他得到的结果是：如果广义黎曼猜想（REH）成立，则有一个算法存在，它对每个n，可在$\log_2 n$的多项式时间内判明n是否素数。即存在素性判别的多项式算法，而且可设计出这个算法。

这个关系的发现，也是以研究费马小定理为出发点的。下面熟知的欧拉的结果是费马小定理的推广。

定理2.11 若n是一个奇素数，则对任意的自然数a，$n\nmid a$有 $a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$，其中$\left(\frac{a}{n}\right)$是勒让德符号。

1976年初，勒默发表了一篇短小精悍的文章，证明了定理1的逆定理也成立，他得到了。

定理2.12 （勒默） 若n是奇合数，则存在自然数a，满足$(a,n)=1$，使得$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\pmod{n}$，这里$\left(\frac{a}{n}\right)$是雅可比符号。

证明 若n含有因子p^{α}，p是奇素数，$\alpha>1$。取a为p^{α}的原根，且由孙子定理，可要求$(a,n)=1$．由于$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$，推出$a^{n-1}\equiv1\pmod{n}$，即$a^{n-1}\equiv1\pmod{p^{\alpha}}$即$\varphi(p^{\alpha})\mid n-1$，即得$p^{\alpha-1}(p-1)\mid n-1$，但$p^{\alpha-1}\mid n$，故这不可能，因此$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\pmod{n}$。

若$n=p_1p_2\cdots p_t$，$t\geqslant2$，p_1，p_2，…，p_t为互不相同的奇素数。由孙子定理，可取a_1，使得$\left(\frac{a_1}{p_1}\right)=-1$而$\left(\frac{a_1}{p_i}\right)=1$，$i=2$，…，t。则$(a_1,n)=1$

且$\left(\frac{a_1}{n}\right)=-1$。下面用反证法，若对每个满足$(a,n)=1$的 a 都有 $a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)(\bmod n)$，则有$a^{n-1}\equiv 1\ (\bmod n)$。取$a_{p_2}$为$p_2$的原根，即得$a_{p_2}^{n-1}\equiv 1\ (\bmod p_2)$，所以 $p_2-1\mid n-1$，即 $\frac{n-1}{p_2-1}$是个整数。现对 a_1，由欧拉定理 2.11 $a_1^{\frac{p_2-1}{2}}\equiv\left(\frac{a_1}{p_2}\right)(\bmod p_2)$ 我们有

$$-1=\left(\frac{a_1}{n}\right)\equiv a_1^{\frac{n-1}{2}}=\left(a_1^{\frac{p_2-1}{2}}\right)^{\frac{n-1}{p_2-1}}\equiv$$

$$\left(\frac{a_1}{p_2}\right)^{\frac{n-1}{p_2-1}}=1^{\frac{n-1}{p_2-1}}=1\ (\bmod p_2)$$ 即$2\equiv 0\ (\bmod p_2)$，

但p_2是奇素数。这是不可能的。故必有a使$(a,n)=1$，$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\ (\bmod n)$。 □

尽管勒默的这个结果说明了，若 n 是合数，则在1到n 之间至少存在一个 a 使 $a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)(\bmod n)$ 成立，但他并没有指出如何去找 a，也没有指明 a 在何处。对某个数 n，当然，我们可以对 1 到n 之间的每个数检验条件 $a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)(\bmod n)$ 是否满足，若满足，则 n 是素数。然

而，对 1 到n之间的每个数a去检验 $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)$ (modn)的计算量是O（$n\log_2^3 n$）。这个计算量太大了。因此，我们希望能够改进勒默的这个结果。如果对合数 n，能确认在较小的范围内（而不是 1 到n之间）存在a使$a^{\frac{n-1}{2}} \not\equiv \left(\frac{a}{n}\right)$ (modn)，那就可以减少检验 $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)$(modn) 的计算量，因为这时只要对较少的 a 检验。这样就可望得出较有效的素性判别法。

缪内注意到，给定一个数 n，在 n 的缩系组成的群$U_n = \{a(\text{mod}n) \mid a \in Z, (a,n) = 1\}$中，满足条件 $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)$(modn)的a (modn)全体构成一个子群，设其为 M_n。于是就可以利用下面的安克尼和蒙特哥梅利的结果。

定理2.13 （安—蒙） 在广义黎曼猜想(REH)成立的条件下。存在一个常数C，对任何自然数n及U_n到任何一个群G 的非单位同态 ψ，都存在q使 $1 < q \leqslant C(\log_2 n)^2$ 且ψ (qmodn) ≠ 1（其中1是G 的单位元）。

由此，缪内得到了下面的结果：

定理2.14 （缪内） 在广义黎曼猜想的成立之前提下，存在一个常数 C，对任何的自然数

n，若n是合数，则存在$1\leqslant a\leqslant C(\log n)^2$使得$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\pmod{n}$。

证明 定义ψ：$\psi(a \bmod n)=\left(\frac{a}{n}\right)a^{\frac{n-1}{2}}\pmod{n}$。则$\psi$显然是$U_n$到$U_n$的同态映射。因为n是合数，由定理2.12知，$\psi$不是单位同态映射。因此，由定理2.13，存在a使$1<a\leqslant C(\log_2 n)^2$使$\psi(a \bmod n)\neq 1 \bmod n$，即$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\pmod{n}$。

注 定理2.14中的常数C可取作与定理2.13中的C相同。

维路于1978年指出，定理2.14中的常数C可以定为70。

由定理2.14，我们说，在广义黎曼猜想成立的前提下，素数判别的多项式算法是存在的。我们可以如下设计出这样一个算法：

对任何输入n，依次对$a=1,2,\cdots,70(\log_2^2 n)$检验$a^{\frac{n-1}{2}}\not\equiv\left(\frac{a}{n}\right)\pmod{n}$是否成立。若对其中的一个a成立，则停止检验，这时n是合数（由定理2.11），若对每一个a都不成立，也就是说，对$a=1,2,\cdots,70(\log_2^2 n)$都有$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$，由定理2.14则n是素数。

上述算法可以完全确定地判别n是否素数，

其计算量就是作 $70(\log_2^2 n)$ 个同余式 $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) (\mathrm{mod}\, n)$ 检验的计算量。用O符号表示，即工作量是 $O(\log_2^5 n)$，因而这是一个多项式算法。

由此可见，只要广义黎曼猜想成立，则素性判别的多项式算法是找到了的。但证明广义黎曼猜想是相当困难的，这是数学家们一直关注的问题。这里的讨论也表明，素性判别是需要用较高深的方法来研究。

6. 一种概率算法

缪内的结果虽然很好，但它毕竟是依赖于一个悬而未决的假设。因而在实用中，它是不能被采用的。故我们回到勒默的结果，看看从这个结果还能引伸出什么方法来。

勒默的结果说，若 n 是合数，则存在 a，满足 $(a,n)=1$ 使 $a^{\frac{n-1}{2}} \not\equiv \left(\frac{a}{n}\right) (\mathrm{mod}\, n)$。但他没说这样的 a 有多少。下面的定理就说明了这样的 a 至少有多少。

定理2.15 若 n 是合数，则在 1 到 n 之间，

至少有$\frac{1}{2}\varnothing(n)$个满足$(a,n)=1$的a使$a^{\frac{n-1}{2}}$

$\not\equiv\left(\frac{a}{n}\right)\pmod{n}$。

证明 U_n中满足$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$的元素构成一个子群$M_n$。当n是合数时，由勒默的结果定理2.12，$M_n$是$U_n$的真子群，即$M_n\neq U_n$，因而$M_n$在$U_n$中的指标至少是2，即$(U_n:M_n)\geqslant 2$。故$M_n$中的元素个数至多是$U_n$中元素个数的一半，即$U_n$中不在$M_n$中的元素个数至少占一半，而$U_n$的元素个数是$\varnothing(n)$，因此，在1到n之间，至少有$\frac{1}{2}\varnothing(n)$个满足$(a,n)=1$的a使$a^{\frac{n-1}{2}}$

$\not\equiv\left(\frac{a}{n}\right)\pmod{n}$.

推论 若n是合数，则1到n之间至少有一半的数a不满足$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$.

证明 对1到n之间的数a，若$(a,n)\neq 1$，则显然a不满足$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$．而对与n互素的a而言，在1到n之间有一半不满足$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right)\pmod{n}$（由定理2.15）．总之，在1到n之间，至少有一半的数a不满足$a^{\frac{n-1}{2}}\equiv$

$\left(\frac{a}{n}\right)$ (modn)。

这个推论可以产生一种作“素性判别”的概率算法：

对任何输入n，从1到n之间随机地抽取k个数$a_1, a_2, \cdots, a_k$（k是事先依所需要的可靠性确定的）。逐个对a_i检验$a_i^{\frac{n-1}{2}} \equiv \left(\frac{a_i}{n}\right)$ (modn)是否成立，若有某个a_i使此同余式不成立，则断言n是合数；若对$a_1, \cdots, a_k$，同余式都成立，则断言n是素数。

在这个算法中，a_i的选取是随机的，而且结论（断言）正确性不是完全确定的，故此算法叫概率算法。在这个概率算法中，当得到断言说输入n是合数时，由定理2.11，结论是正确的；当得到断言说输入是素数时，没有什么定理可以确保结论是正确的。也就是说，此算法在执行完毕后，可能将一个事实上是合数的输入断言为是素数了。但是，由以上定理2.15的推论，这种出错的概率是很小的。因为，若n事实上是合数，则随机地于1到n之间选取一个a，a满足$a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)$ (modn)的概率小于$\frac{1}{2}$，因而对随机选取的$a_1, a_2, \cdots, a_k$，同余式$a_i^{\frac{n-1}{2}} \equiv \left(\frac{a_i}{n}\right)$ (modn)都

成立的概率就小于$\frac{1}{2^k}$。于是，上述算法将事实上的合数断言为素数这种出错的概率小于$\frac{1}{2^k}$。当k取得很大时，例如k=1000，则出错的概率几乎为零（但不是零！），因此用此算法作素数判别2^{1000}次，仅可能出现一次错误（从统计的角度来说）。然而，对某个特定的n，通过算法后被断言为素数，我们并不能确定这个断言一定不是错误的。因为以上的原因，这个算法常被称为“合数判别”算法而不称为素数判别算法。但是，因为这个算法的计算量很小，只有$O(k\log_2^3 n)$，故它确实常被采用。例如用它来寻找一组数，使其中绝大多数是素数。最近，还有人（如赖宾）用它来支持一些孪生素数和素数分布的猜想。

7. 目前最有效的艾德利曼—鲁梅利算法

1983年，素数判别的研究里，出现了突破性的进展，那就是艾德利曼和鲁梅尼提出来了一个方法。这个方法是一个近似多项式算法，它在实际应用中，对一千位以下的数而言，与多项式算法一样有效。

要系统地讲述这个算法，需要涉及到代数数论、乘法数论及多项式理论等多方面的专门知识。因而，我们这里只是概要地介绍这个算法，很多证明就略去了。但我们尽量保证读者能从介绍中对算法的基本思想、方法和技巧有个大概了解。我们将介绍的不是艾德利曼—鲁梅利原来的形式，而是由勒恩斯爵简化了的形式。

设p，q是两个素数，$p\mid q-1$，记ξ_p和ξ_q分别为p次和q次本原单位根。设g_q是q的原根，即g_q是群U_q的生成元。定义映射$X_{pq}: U_q \to \langle\xi_p\rangle$（其中$\langle\xi_p\rangle$为由$\xi_p$生成的循环群）为$X_{pq}(g_q^j)=\xi_p^j$。因为$p\mid q-1$且$U_q$是$q-1$元循环群。则$X_{pq}$是同态映射。对于任给的整数t，定义高斯和$\tau(X^t_{pq})=\sum\limits_{u\in U_q} X_{pq}(u)^t\xi^u_q$。因此$\tau(X^t_{pq})$是$R_{pq}=Z[\xi_p,\xi_q]$中的元素。（其中$R_{pq}=Z[\xi_p,\xi_q]=\{\sum\limits_{i,j}a_{ij}\xi_p^i\xi_q^j \mid a_{ij}\in Z\}$是一个环）

由高斯和的性质，有：若n是素数，则$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n)\pmod n$（注意：因为$\tau(X_{pq})$是$R_{pq}$中的元素，因而，我们是在环$R_{pq}$中讨论同余式的）。易知，$p=2$时，这就是费马小定理，因此，上述的素数的这个性质，事实上就是费马小定理在环R_{pq}中的推广。我们来看看，当一个数n，对一些p，q，$p\mid q-1$满足$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n)\pmod n$时，能对n作出什

么结论。勒恩斯爵对此给予了一个回答，他证明了下面的定理。

定理2.16 设P是一个素数集合，设$\prod_{p\in P} p = Z$。Q是由P产生的另一个集合满足，对任意的一个素数$q\in Q$有$q-1\mid Z$。若奇数n满足下列条件：

(i) 对每个$p\in P$，$q\in Q$，若$p\mid q-1$，则$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n) \pmod n$。

(ii) 对每个$p\in P$和n的素因子r有$p^{e_p}\mid r^{p-1}-1$，其中$p^{e_p}\parallel n^{p-1}-1$。

则每个n的素因子r，有$r\equiv n^i \pmod w$，其中i有$0\leqslant i<Z$，$w=\prod_{q\in Q} q$。

注 定理2.16中的条件(ii)看起来似乎难以验证，而事实上，可以证明，若对某个p，存在q（q不必在Q中）使$p\mid q-1$且$X_{pq}(n)\neq 1$，$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n) \pmod n$，则条件(ii)就对p及任何n的素因子r成立。在实际应用时，对某个p要使$X_{pq}(n)\neq 1$对某个素数q（$p\mid q-1$）成立是很容易做得到的。

由勒恩斯爵的结果，我们可以如下编出一个算法，这个算法是基于艾德利曼—鲁梅利的思想，采用了勒恩斯爵的表述。

对任何输入n，n是奇数。第一步，寻找素数集合P，使得由其产生的素数集Q满足$W=\prod_{q\in Q} q>\sqrt{n}$。第二步，施行两子步：

（Ⅰ） 对每个$p\in P, q\in Q$使$p\mid q-1$，验证$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n)\ (\mathrm{mod}\, n)$（即定理2.16中的条件（i）

（Ⅱ） 对每个$p\in P$，寻找一个q（不必属于Q）使$p\mid q-1$且$X_{pq}(n)\neq 1$。（这是为了让定理2.16中的条件（ii）成立。）

如果（Ⅰ）对某对$p\in P$，$q\in Q$（$p\mid q-1$）不成立或（Ⅱ）对某个$p\in P$不存在q使$p\mid q-1$，$X_{pq}(n)\neq 1$，$\tau(X_{pq})^{n^{p-1}-1}\equiv X_{pq}(n)\ (\mathrm{mod}\, n)$，则$n$是合数。否则，即（Ⅰ），（Ⅱ）都通过了，由定理2.16知，n的任何一个素因子r满足$r\equiv n^i\ (\mathrm{mod}\, w)$，其中$1\leqslant i<\prod_{p\in P} p=Z$。于是执行下面的第三步：

对$r=n^i \mathrm{mod}\, w$，$i=1, 2, \cdots, Z-1$，逐个检验r是否整除n。若有其中之一$r\neq 1$，n使$r\mid n$，则n是合数，否则，由$w>\sqrt{n}$，则n是素数。

在上述算法中，第二步的计算量，可用高斯和的性质（它所满足的最小多项式）估计出来，它是$\log_2 n$的多项式。第一步和第三步可以在$O[(\log_2 n)^{C\cdot\log_2\log_2\log_2 n}]$的计算量内完成，这个由下面的波门伦斯的结果得出。

定理2.17 设P, Q如上定义，$Z=\prod_{p\in P} p$，

$w=\prod\limits_{q\in Q} q$，则存在绝对可计算的常数$C_1, C_2>0$，使得，当$w>n^{\frac{1}{2}}\geqslant 10$时，Z的最小可能值满足。

$$(\log_2 n)^{C_1\log_2\log_2\log_2 n}<Z$$
$$<(\log_2 n)^{C_2\cdot\log_2\log_2\log_2 n}$$

在第三步中，至多要验证Z个整除式，每个整除式的计算量是$O(\log_2^3 n)$，因而，完成第三步至多需要$O[(\log_2 n)^{C\log_2\log_2\log_2 n}]$的计算量。

总之，用上述算法验证一个数n是否素数，几乎只需要多项式时间。因为当n的位数不太大时（例如少于1000位），$C(\log_2\log_2\log_2 n)$是很小的，故其计算量还不及一个多项式算法多。

8. 一些特殊的素数及其判别

麦森勒素数

所谓麦森勒数是指形如2^p-1（p是素数）的数，记为M_p，$M_2=3$，$M_3=7$，$M_5=31$，$M_7=127$等，麦森勒素数即是麦森勒数又是素数者。

早在1644年，麦森勒就对p＝2，3，5，7，11，13，17，19计算了M_p，他证明了除p＝11外，其它的M_p是素数，他由此断言，不大于257的各素数，只有p＝2，3，5，7，13，17，19，31，67，

127，257使M_p是素数。当时没有谁（包括他本人）证明了这个断言。直到1772年，欧拉经过多年探索，证明了是$2^{31}-1=M_{31}$是素数，大约在1875年，努卡斯证明了$2^{127}-1$是素数，但是证明了M_{67}不是素数。因此，麦森勒的断言就不全对了。1886年，有人证明了$2^{61}-1$是素数，因而，人们怀疑麦森勒在抄写时，将61误抄成了67。然而，1911年，泡尔斯证明了$2^{89}-1$也是素数，三年后，又证明了$2^{107}-1$也是素数。最后，1922年，葛莱启克证明了$2^{257}-1$不是素数。这样就彻底说明了麦森勒的断言是不对的。但麦森勒的断言激发了人们对麦森勒素数的研究。到目前为止，已经得到的前29个麦森勒素数M_p是：

p＝2，3，5，7，13，17，19，31，61，89，107，127，521，607，1279，2203，2281，3217，4253，4423，9689，9941，11213，19937，21701，23209，44497，86243，132049。

M_{132049}是一个39751位的素数，是目前已知的最大的素数。要对诸如M_{132049}这么大的数判别其是否素数，肯定要用到数的特殊形式，用特殊的方法（艾德利曼—鲁梅利方法也无济于事了）。另外，从上面看出，那些使M_p为素数的素数p的分布是不规则的。因而有必要对麦森勒数研究其

特殊的素性判别方法。有一种目前最有效的、专门用于麦森勒数的素性检验方法是下述的努卡斯一勒默方法。

定理2.18 设p是一个奇素数，定义序列$\{L_n\}$如下：

$$L_0=4,\quad L_{n+1}=(L_n^2-2)\ \mathrm{mod}\ 2^p-1$$

则$M_p=2^p-1$是素数的充分必要条件是$L_{p-2}=0$。

证明定理2.18之前，需要罗列一下努卡斯序列

$$U_0=0,U_1=1,U_{n+1}=4U_n-U_{n-1}$$

$$V_0=2,V_1=4,\quad V_{n+1}=4V_n-V_{n-1}$$

的一些性质：

(i) $V_n=U_{n+1}-U_{n-1}$

(ii) $U_n=\dfrac{((2+\sqrt{3})^n-(2-\sqrt{3})^n)}{\sqrt{12}}$

(iii) $V_n=(2+\sqrt{3})^n+(2-\sqrt{3})^n$

(iv) $U_{m+n}=U_mU_{n+1}-U_{m-1}U_n$

(v) $U_{2n}=U_nV_n,\ V_{2n}=V_n^2-2$

(vi) 若$U_n\equiv0\pmod{p^e}$，则$U_{np}\equiv0\pmod{p^{e+1}}$（p是任何一个素数）

(vii) $U_{p-\varepsilon(p)}\equiv0\pmod{p}$，$\varepsilon(p)=\left(\dfrac{3}{p}\right)$，

p是素数。

(viii) 对任一个整数m，定义m对$\{U_n\}$的

幻秩为使$U_n \equiv 0 \pmod{m}$成立的正整数 n 中的最小者。记为$\rho(m)$，则有

$U_n \equiv 0 \pmod{m}$的充分必要条件是$\rho(m) \mid n$。

以上这些性质都可由努卡斯序列的定义用归纳法得到[1]。

现在我们来证明定理 2.18。

由$\{L_n\}$的定义及(v)易知：$L_n \equiv V_{2^n} \pmod{2^p-1}$。又由$2U_{n+1} = 4U_n + V_n$及$\gcd(U_{n+1}, U_n) = 1$得$\gcd(U_n, V_n) \mid 2$。进而知，$U_n$和$V_n$无公共的奇因子。如果$L_{p-2} = 0$，我们必有

$$U_{2^{p-1}} = U_{2^{p-2}} V_{2^{p-2}} \equiv 0 \pmod{2^p-1}$$

$$U_{2^{p-2}} \not\equiv 0 \pmod{2^p-1}$$

又，如果$\rho = \rho(2^p-1)$是2^p-1对$\{U_n\}$的幻秩，则$\rho \mid 2^{p-1}$但$\rho \nmid 2^{p-2}$，故$\rho = 2^{p-1}$。由此，我们推得$n = 2^p - 1$是素数。否则，设$n = p_1^{e_1} \cdots p_r^{e_r}$是n的标准分解式，则$p_i > 3$〔因为$n = 2^p - 1 \equiv (-1)^p - 1 \equiv -2 \pmod{3}$〕$i = 1, \cdots, \gamma$，令$t = \mathrm{Lcm}\,[p_1^{e_1-1}(p_1-\varepsilon_1), \cdots, p_r^{e_r-1}(p_r-\varepsilon_r)]$，其中$\varepsilon_i = \left(\frac{3}{p_i}\right)$，则由 (vi)，(vii)，(viii)得$U_t \equiv 0 \pmod{2^p-1}$。又由(viii)得$2^{p-1} \mid t$。令$n_0 = \prod_{1 \le j \le r} p_j^{e_j-1}(p_j - \varepsilon_j)$，则有$n_0 < \prod_{1 \le j \le r} p_j^{e_j-1}\left(p_j + \frac{1}{5}p_j\right) = \left(\frac{6}{5}\right)^r \cdot n$。因为诸$p_i - \varepsilon_i$是偶数，则$t \le \frac{n_0}{2^{r-1}}$（在取最小公

倍数时，有 $r-1$ 个 2 因子因重复而不取），综上所述有：

$$\rho \leqslant t \leqslant 2\left(\frac{3}{5}\right)^r n < 4\left(\frac{3}{5}\right)^r \rho < 3\rho$$

因此 $r \leqslant 2$ 且 $t=\rho$ 或 2ρ，故 t 是 2 的幂数。因此诸 $e_i=1$，这时，若 $r=2$，则 $n=2^p-1=(2^k \pm 1)(2^l \pm 1)$，其中 $2^k \pm 1$ 和 $2^l \mp 1$ 是素数，这是不可能的。因而只有 $r=1$，即 n 是素数。

反过来，若 $n=2^p-1$ 是素数，由 $L_n=V_{2n} \bmod 2^p-1$，直接验证，即知 $L_{2^{p-2}}=0$。

到目前为止，人们得到的最大的素数 M_{132049} 就是用这个努卡斯—勒默方法得到的，这是1983年9月由斯诺文斯基在大量计算后得到的。

近期，威廉斯对努卡斯—勒默的方法作了推广，他得到一些用于对诸如 $10^n \pm 10^m + 1$，$a^n-1/a-1$ 等形式的数作素性判别的方法。

费马素数

形如 $2^{2^n}+1$ 的数称为费马数，记为 F_n，若它又是素数，则称为费马素数。

早在17世纪，费马验证了 $F_0=3$，$F_1=5$，$F_2=17$，$F_3=257$，$F_4=65537$ 是素数。据此，他猜测 F_n 都是素数。但到1732年，欧拉分解了 F_5：

$$F_5=2^{2^5}+1=641\times 6700417$$

故费马的这个猜测不成立。自此之后，人们对

$n = 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 23, 25, 26, 27, 29, 30, 32, 36, 38, 39, 42, 52, 55, 58, 62, 63, 66, 71, 73, 75, 77, 81, 91, 93, 99, 117, 125, 144, 147, 150, 201, 205, 207, 215, 226, 228, 250, 255, 267, 268, 275, 284, 287, 298, 316, 329, 334, 398, 416, 452, 544, 556, 637, 692, 744, 931, 1551, 1945, 2023, 2089, 2456, 3310, 4724, 6537, 6835, 9428, 9448, 23471$证明了$F_n$是合数。然而，除前5个费马数为素数外，再也没发现任何费马素数了。因而人们更倾向于认为费马素数只有有限个，但没人对此作出证明。近年来，人们把精力放在分解费马数上，没人研究特殊的判别法来对费马数作素性判别。

$k \cdot 2^n + 1$ 型素数

研究$k \cdot 2^m + 1$型素数，对分解费马数有着重要作用。熟知，费马数F_n的每个素因子都具有形状$k \cdot 2^m + 1$，其中$m \geqslant n + 2$，k是奇数。当已知某些$k \cdot 2^m + 1$是素数时，用它去试除F_n（$n \leqslant m - 2$），这样就可以找到一些费马数的因子。另外，在验证了某个$k \cdot 2^m + 1$是素数后，再对$k \cdot 2^m - 1$作素性判别，则可能找出一对孪生素数来，在这个方面的研究是从普罗丝开始的，他首先提出了一个判别$k \cdot 2^m + 1$（$k < 2^m$）型数是否素数的方法。

定理2.19 给定$N=k\cdot 2^m+1$，$k<2^m$，先寻找一个整数D使得雅可比符号$\left(\frac{D}{N}\right)=-1$（若$3\nmid k$时，可取$D=3$，而$3\mid k$时，不难找出一个合适的D)，则N是素数的充分必要条件是$D^{(N-1)/2}\equiv -1\pmod N$。

用普罗丝的这个结果及一些具体的方法，贝利、鲁宾逊、威廉斯等人对一些奇数k和m决定的数$k\cdot 2^m+1$作了系统的考察，他们对$1\leqslant k\leqslant 150$，$1\leqslant m\leqslant 1500$，找出了所有$k\cdot 2^m+1$型的素数。又对$3\leqslant k\leqslant 29$和$1500<m\leqslant 4000$，列出了所有的$k\cdot 2^m+1$型的素数，得出了7个新的费马数的因子，而且还得到了一对很大的孪生素数：$297\cdot 2^{546}+1$和$297\cdot 2^{546}-1$。(目前发现的最大的孪生素数是$156\cdot 5^{202}\pm 1$)

最后，介绍斯塔克的一个结果。若m固定，序列$\{a_k\}=\{k\cdot 2^m+1\}$含有无限多个素数，这是由狄利克雷定理得出的。自然地，人们提出这样一个问题：当k固定时，序列$\{b_m\}=\{k\cdot 2^m+1\}$是否含有无限多个素数呢？斯塔克对这个问题给了否定的回答，他给了一个$k=29353633315419255 31$，它使序列$\{k\cdot 2^m+1=b_m\}$中每一项都是合数。

由1组成的素数

每位数都是1的n位数记为R_n，例如$R_1=1$，$R_2=11$，$R_7=1111111$等等。熟知$R_2=11$是素数。

因为当$m|n$时，$R_m|R_n$，故R_n是素数的必要条件是n是素数。然而，n是素数时，R_n不一定是素数，例如$n=3$时，$R_3=111=3\times37$不是素数。至今为止，只有得出当$n=2,19,23,317,1031$时，R_n是素数。其中R_{19}已由前面的"$n\pm1$"判别法证明了，R_{23}是1929年被证明为素数的。R_{317}是1979年由威廉斯证明的，R_{1031}是1986年由威廉斯和达内用第四节介绍的方法证明为素数的。这五个素数是R_n系统中的前五个素数，即其它不大于1031的素数n都使R_n为合数。事实上，达内还证明了，第六个R_-素数必定大于R_{10000}，也就是说，在1到R_{10000}之间只有以上五个R_-素数。

9. 在计算机上实施素数判别的战略

迄今为止，所有得到的素性判别法有简单明了的试除法，有赖宾等人的概率算法，有努卡斯到威廉斯的利用$n\pm1$，n^2+1，$n^2\pm n+1$的因子方法，还有艾德利曼—鲁梅利的最有效的近似多项式方法。这些方法各有利弊，它们各适用于适当

的输入。上述第一种算法虽然很耗费时间（对n作素性判别的计算量是O（$\sqrt{n}\log_2^2 n$），但它很简单，并且直接了当，对较小的数n（最大不超过10^8）合适。第二种算法，用于作素性判别时，它可能出现差错，但出差错的可能性很小，因而可用它来确认哪些数最有可能是素数。努卡斯和威廉斯的$n\pm1$，n^2+1，$n^2\pm n+1$方法，对于20—50位的素数是有效的工具。艾德利曼—鲁梅利的方法是普遍的方法，它不依赖于待判别的数的特殊性，而且计算量是$O((\log_2 n)^{C\log_2\log_2\log_2 n})$，故一般利用它来对50位以上的数来作素性判别。我们摘抄波门伦斯的一个表，它表明了三种方法的优劣。

方　法	20位	50位	100位	200位	1000位
试除法	2小时	10^{11}年	10^{36}年	10^{86}年	10^{486}年
努卡斯和威廉斯的方法	5秒	10小时	100年	10^9年	10^{44}年
艾德利曼和鲁梅利的方法	10秒	15秒	40秒	10分	1周

另外，对有些特殊的数作素数判别时，用特殊的方法（如对2^p-1，用勒默的方法）就可能很便利，就不必用艾德利曼—鲁梅利的方法。下面，我们指出在计算机上作素性判别的三大步

骤。

对于一个待判定的数 n（设 n 不具有我们了解的特殊形式），实行下面三步：

（1） 先用 1 到1000之间的素数（它们通常储存在计算机内）去试除。若 n 恰好有小的素因子，则不是素数，算法可停止；否则，进行下一步。

（2） 对n随机地选取一些底作伪素数检验，（即计算 $a^{\frac{n-1}{2}} \bmod n$，并看其是否等于$\left(\frac{a}{n}\right)$。若n 不通过这些检验，则 n 是合数；否则，我们得出结论说n 是素数。下面的步骤就是证明 n 确为素数。

（3） 若n 具有20—50位以下，可使用努卡斯和威廉斯的方法。

若n 具有30—1000位或以上，即使用艾德利曼和鲁梅利的方法。

注 对 30—50 位的数，用努卡斯和威廉斯的方法，还是用艾德利曼—鲁梅利的方法，要视计算机的性能和实际情况而定.

三 大数分解



大数分解是与素性判别紧密相关的课题。对于给定的一个自然数，先对它作素性判别。若它是素数则罢，若它是合数，我们还想知道它的因子分解式。历史上及现在都有这样的数，它被判别为合数，但是没有发现它的因子（即没有分解）。例如，F_8 在1909年就被证明为合数，但直到1975年，才发现它的一个因子。又如，F_{14} 早在1963年就被证明为合数，但至今它的因子还不知是什么。因此，我们要系统地研究如何去分解一个已知是合数的数。下面，我们假定下面待分解的数是合数。为了方便起见，还假定，对待分解的数已经用试除法排除了它在1到10000（或10^5）范围内的因子。（这在计算机上是非常容易实行的事，因为1到10^5之间的素数和试除法都可以存储在计算机内。）

与素性判别法的产生一样，大数分解的方法的产生也是从注意合数的一些性质开始的。因而，我们要对合数的一些性质（特别是结构方面的性质）作考察和研究，由此引伸分解的方

法。

1. 经典的方法

费马方法

费马注意到，给定 n，若 n 是两数平方差 $n=a^2-b^2$，则 $n=(a+b)(a-b)$ 是 n 的一个因子分解．于是，我们要设法将 n 表为两个平方差。用逐个考察方法，从 $b=1$ 开始，依次考察 $n+b^2$ 是否平方数，若恰对某个 b，有 $n+b^2$ 是平方数，设为 a^2，即 $n+b^2=a^2$，故 $n=(a+b)(a-b)$。

用这个方法，只有当n有两个几乎相等的因子时，才比较快。因为当有两个因子 $a+b$，$a-b$ 几乎相等时，$b=\frac{1}{2}[(a+b)-(a-b)]$ 就很小，即从1开始作逐个试验的次数就少．另外，我们可以不须对所有的 b 来计算$n+b^2$ 并考察它是否平方数。我们注意到，若 $n+b^2$ 是平方数，则 b 的奇素因子 p 一定满足$\left(\frac{n}{p}\right)=1$ 或 0（其中$\left(\frac{n}{p}\right)$ 是勒让德符号），故只须对不含使$\left(\frac{n}{p}\right)=-1$ 的奇素因子 p 的 b 进行考察即可．当 n

恰有两个几乎相等（相对于n而言，相差 $\log_2 n$ 的常数倍数即算作几乎相等）时，用费马方法去分解n是很有效的。

例 分解 385374826089807。

设此数为 n。因为 $\left(\frac{n}{5}\right)=\left(\frac{2}{5}\right)=-1$，

$\left(\frac{n}{7}\right)=\left(\frac{1}{7}\right)=1$，$\left(\frac{n}{11}\right)=\left(\frac{1}{11}\right)=+1$，

$\left(\frac{n}{13}\right)=\left(\frac{8}{13}\right)=-1$，又 $n\equiv 3 \pmod 4$，故

$2\nmid b$，因而只要对 $b=3,7,9,11,17,19,21,23,27,29,31,33,37,41,43,47,49,51,53,57,\cdots\cdots$，试验 $n+b^2$ 是否为平方数，结果发现 $n+57^2=19630966^2$，故 $n=19630909\times 19631023$。

一种筛法

对于给定的n，若已知n的一些较小的平方剩余，则就可以筛去一些 n 的不可能的因子，因而缩小了n的可能因子的范围，再进一步对它们进行试除。譬如，若a是n的一个二次剩余，则对每个n的奇素因子p，有 $\left(\frac{a}{p}\right)=1$，由二次互倒律，则p在一些差为a（当 $4\mid a-1$ 时）或4a（当 $4\nmid a-1$ 时）的等差序列中，因而，只在这

些等差序列中找 n 的素因即可。用这个方法，可以得到下面的一个有用的定理（前面第二章第八节中谈费马素数时曾提到这个定理）。

定理 3.1 费马数$F_n = 2^{2^n}+1 (n\geqslant 2)$的素因子都具有形状$2^{n+2}\cdot k+1$。

证明 设$p \mid F_n$，则$2^{2^n}+1\equiv 0 \pmod p$

即$2^{2^{n+1}}\equiv 1 \pmod p$。故 2 对模 p 的次数为$2^{n+1}$，因而，$2^{n+1} \mid p-1$。故$p=1+2^{n+1}\cdot k$对某个 k 成立。当$n\geqslant 2$时，$p\equiv 1 \pmod 8$，因此 2 是 p 的二次剩余，因而$2^{\frac{p-1}{2}}\equiv 1 \pmod p$，故

$2^{n+1} \left| \frac{p-1}{2}\right.$，即$2^{n+2} \mid p-1$。所以，存在 k 使

$p=2^{n+2}\cdot k+1$。 □

当 n 的较小的二次剩余容易找到时（例如 2，3，5 等是 n 的二次剩余时），可用筛法去分解 n，我们举一简单的例子，说明这个方法的用途。

例 分解1711。

因为$16^2\cdot 7-9^2=1711$，$37^2\cdot 5-1=4\cdot 1711$，故 7 和 5 都是1711的二次剩余。因而，对1711的任意一个素因子 p 有$\left(\frac{7}{p}\right)=1$，$\left(\frac{5}{p}\right)=1$，

因此$p\equiv \pm 1, \pm 9, \pm 25 \pmod{28}$，$p\equiv \pm 1$

(mod 5)。因为满足这两组同余式的前几个数是1，9，19，29，…，经对它们依次试除1711知29¦1711，故$1711=29\cdot59$。

勒让德方法

勒让德首先注意到，若一个奇数n是合数且至少有两个不同的素因子，则$x^2\equiv a^2 \pmod n$至少有四个解。其中$x\equiv\pm a \pmod n$是两个解，这两个解称为平凡解，另外的解就称为非平凡解。若对同余式$x^2\equiv a^2 \pmod n$，能找到一个非平凡解b，则$\gcd(b+a,n)$或$\gcd(b-a,n)$都是n的真因子。因而，分解n就等价于要寻找形如$x^2\equiv a^2 \pmod n$的同余式的非平凡解。这便是以下的连分数方法和二次筛法所要解决的问题。

2. 蒙特卡罗方法

设S是一个有限集，令Map (S)是S到S的映射全体。对于每一个$\psi\in$ Map (S)和$x_0\in S$，由于S是有限集，则存在$l\geqslant1$使$x_0=\psi^0(x_0)$，$\psi(x_0)$，…，$\psi^{l-1}(x_0)$（ψ^i表示ψ的i次复合）互不相同。但$\psi^l(x_0)$与某个$\psi^{l'}(x_0)$（$0\leqslant l'\leqslant l-1$）相同，这时，称l为$x_0$在$\psi$下的轨迹长度。当$k\geqslant l'$时，序列$\{\psi^k(x_0)\}$是一个周期为$d=l-l'$的循环序列。

设S有m个元素。$x_0 \in S$，则 x_0 在 ψ 下的轨迹长度至少是 l 的充分必要条件是 $x_0, \psi(x_0)$，…，$\psi^{l-1}(x_0)$ 互不相同。因此，Map (S) 中使得 x_0 在其下的轨迹长至少是 l 的映射个数是 $m^{m-l} \cdot \prod_{i=0}^{l-1}(m-i)$ 而 $m^{m-l} \cdot \prod_{i=0}^{l-1}(m-i) < m^m e^{-\sum_{i=0}^{l-1}\frac{i}{m}} \leqslant m^m e^{-\frac{l(l-1)}{2m}}$，因此，对任意的 $\lambda > 0$，在Map (S) 中，使得 x_0 在其下的轨迹长至少是 $(2\lambda m)^{1/2}+1$ 的映射占的比例小于 $e^{-\lambda}$。故，只有很少的对 (x_0, ψ) 使得 x_0 在 ψ 下的轨迹长很大（譬如大于 $5m^{1/2}$ 的对只占 $e^{-12.5} \approx 0.0000034$），我们称使得 x_0 在 ψ 下的轨迹长很大（譬如大于 $5m^{1/2}$）的对 (x_0, ψ) 叫"例外"。

波纳德就是用上述的事实得到他的分解方法的。假定 n 是合数，设 r 是 n 的一个因子，令 $S = Z_r$ 为模 r 的完全剩余系，设 x_0 是一个整数，ψ 是一个整系数多项式，若 (x_0, ψ) 不是个"例外"，则 x_0 在 ψ 下的轨迹长不大于 $r^{1/2}$ 的一个较小的倍数（譬如5倍）。不过，我们还不知道 r，故不能计算 $x_k = \psi^k(x_0 \bmod r)$，但是，我们可以计算序列 $\{y_k\}$：$y_1 = \psi(x_0) \bmod n$，$y_k = \psi(y_{k-1}) \bmod n, k \geqslant 1$。序列 $\{y_k\}$ 满足 $y_k \equiv x_k \pmod r$，

$k=0,1,2,\cdots$.

由于 x_0 在 ψ 下的轨迹长小于 $r^{\frac{1}{2}}$ 的一个较小的倍数，故当 k 适当大时，$\{x_k\}$ 是一个周期序列，其周期 $l \leqslant cr^{1/2}$（c 比较小），因而，令 $G_k \equiv \prod_{j=1}^{k}(y_{2j}-y_j)\ (\mathrm{mod}\, n)$ 时，必有某些 $k < cr^{\frac{1}{2}}$ 使 $G_k \equiv 0\ (\mathrm{mod}\, r)$ 故 $r \mid \gcd(G_k, n)$，即 $\gcd(G_k, n)$ 是一个 n 的大于 1 的因子。若幸运的话，$\gcd(G_k, n)$ 就是 n 的一个真因子，即 n 被分解了。若对某个 k，$\gcd(G_k, n)=n$，则取另外的 k 试验；若万一对所有的 k 都有 $\gcd(G_k, n)=n$，则从新开始选择 x_0，ψ_0，重复这样的过程，直到 n 被分出一个真因子为止。

以上，我们只知道有些 $k < cr^{1/2}$ 使 $\gcd(G_k, n)$ 可能产生 n 的真因子，但我们不知道哪些 k。不过，我们知道，若 $G_k \equiv 0\ (\mathrm{mod}\, r)$，则 $G_{k+1} \equiv 0\ (\mathrm{mod}\, r)$，因而，从某个 $k=k_0$ 开始，之后的 k 都有 $\gcd(G_k, n)>1$，因此在寻找因子时，不必依次对 $k=1, 2, 3, \cdots$ 来试验 $\gcd(G_k, n)$，而只需要有规则地取一些 k，（如取 $k=1, 2, 4, 8, 16, 32, 64, 128, \cdots$ 或 $k=1, 5, 20, 50, 100, 200, 400, \cdots$.）来看 $\gcd(G_k, n)$ 是否产生 n 的真因子。

上面的 $\{G_k\}$ 是波纳德原先用来寻找 n 的

因子时作出的序列，后来布勒恩特改进了波纳德原先的积式，他用

$$Q_k = \prod_{i=0}^{1+[\log_2(k-1)]} \prod_{j=3\cdot 2^{i-1}+1}^{\min(k,2^{i+1})} (x_{2^i+j} - x_{2^i}) \bmod n$$

取代 G_k，从而使这种方法的效率提高了 25%，即计算时间缩短了25%。

虽然上面已经说明，很少有对 (x_0, ψ) 使 x_0 在 ψ 下的轨迹长很大，但是，对于特定 ψ，有可能对几乎所有的 x_0，都使 (x_0, ψ) 成为“例外”，可以证明，当 $S = Z_r$ 时，ψ 为一次线性多项式和 $\psi(x) = x^2 - 2$ 时，对所有的 $x_0 \in Z_r$ 都产生很多的“例外”来。另外，一方面，我们不能取 ψ 为很高次的多项式，否则，计算 $\{y_k\}$ 时，就要花费很大的计算量。经验表明，我们可以把 $\psi(x)$ 取成二次多项式 $x^2 + a$（其中 $a \neq 2$，$a \in Z$），事实上，在实际中，我们总是用这个多项式。

当 (x_0, ψ) 不是“例外”时，我们在 $k < c r^{\frac{1}{2}}$ 内就可以利用 G_k（或 Q_k）得到 n 的因子，因而这个算法的计算量是 $O(r^{\frac{1}{2}})$，而 n 必有一个小于 $\sqrt{n}$ 的因子，因此，计算量是 $O(n^{\frac{1}{4}})$。

波纳德算法的最大成功是分解了78位的第8个费马数 $F_8=2^{2^8}+1$，它在1909年就证明为合数。他得到了下列分解式

$$F_8 = 1238926361552897$$
$$\times 93461639715357977769163558199606896584051237541638188580280321$$

此外，用波纳德的方法还得到下列数的因子：

$$612053256358933 \mid 10^{103}+1$$

$$150220315444217 \mid 5^{92}+1$$

$$9906434529663163 \mid 7^{71}-1$$

$$441651480271681 \mid 10^{137}+1$$

$$220867654173968270 57 \mid 2^{433}-1$$

$$1225517527330030555 43 \mid 2^{439}-1$$

$$35822885644177092 7 \mid 2^{461}-1$$

$$7320828330474490130 3 \mid 2^{587}-1$$

3. 连分数法

我们知道，利用勒让德方法的一个大困难就

是寻找形如 $x^2 \equiv a^2 (\mathrm{mod}\, n)$ 的同余式的非平凡解。如果我们可以找到一系列二次剩余：$a_1^2 \equiv c_1 (\mathrm{mod}\, n)$，…，$a_k^2 \equiv c_k (\mathrm{mod}\, n)$，而且恰巧 $c_1 \cdots c_k$ 是一个完全平方数。设 $a = a_1 \cdots a_k$，$b^2 = c_1 \cdots c_k$，则有 $a^2 \equiv b^2 (\mathrm{mod}\, n)$，若 $a \not\equiv \pm b (\mathrm{mod}\, n)$，则就找到了同余式 $x^2 \equiv a^2 (\mathrm{mod}\, n)$ 的一个非平凡解。但这里又出现两个困难：其一，怎么产生这么多的二次剩余呢？其二，哪些二次剩余之积恰好是个平方数？

到1931年，勒默和泡尔斯首次用连分数解决了这两个困难，但是，因为当时还没有电子计算机，计算速度的缓慢使得这个连分数方法不被人们注意，直到1975年，莫利桑和卜利尔哈特，对勒默的方法作了深入的研究，将其发展成为一个较系统的好算法，并用此方法，在计算机上成功地分解了屡功不克的 $F_7 = 2^{2^7} + 1 = 2^{128} + 1$，从此以后，连分数法就被人们广泛应用于分解因子。到目前为止，它被认为是最有力的分解工具之一。用它可以方便地在计算机上分解50位左右的数。下面，我们对此方法作一介绍。

由连分数的简单性质知：若 $\sqrt{kn}$ 的连分数展开式为

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\ddots+\cfrac{1}{a_m+\ddots}}}$$

令$\frac{A_m}{B_m}$为它的m次渐近分数，这时，有$A_m^2-knB_m^2=(-1)^{m+1}Q_m$，其中$Q_m$可由一个可以简单计算的递归关系得出。因为对每个m，$A_m^2\equiv(-1)^{m+1}Q_{m+1}(\mathrm{mod}\,n)$，即$\{(-1)^{m+1}Q_{m+1}\}$就是一列二次剩余(modn)，所以，这样就解决了第一个问题。

现在来讨论第二个问题的解决方法。从上列模n的二次剩余$\{(-1)^{m+1}Q_{m+1}\}$中，哪些二次剩余可以选出来成为一个集合使得其元素之积正好是一个完全平方数，从而得到形如$x^2\equiv a^2(\mathrm{mod}\,n)$的一个非平凡解。我们先看一个例子。

例 分解 $n=12007001$。

对$\sqrt{n}$展成连分数，我们得到下表

m	0	11	27	33	40	…
$(-1)^{m+1}Q_{m+1}$	$-2^3\cdot97$	$2^4\cdot71$	2^{11}	$31\cdot97$	$-31\cdot71$	…

这时将第0，11，27，33，40个同余式

$A_0^2\equiv Q_1(-1)\ (\mathrm{mod}\,n)$，$A_{11}^2\equiv Q_{12}(\mathrm{mod}\,n)$，

$A_{27}^2\equiv Q_{28}(\mathrm{mod}\,n)$，$A_{33}^2\equiv Q_{34}(\mathrm{mod}\,n)$，

$A_{40}^2 \equiv (-1)Q_{41} \pmod n$ 乘起来，得到 $(A_0A_{11}A_{27}A_{33}A_{40})^2 \equiv 2^{18}\cdot 31^2\cdot 71^2\cdot 97^2 \equiv (2^9\cdot 31\cdot 71\cdot 97)^2 \pmod n$。另一方面，用连分数的渐近分数的递推公式，可以计算出 A_0，A_{11}，A_{27}，A_{33}，A_{40}，从而可以算出 $A_0A_{11}A_{27}A_{33}A_{40} \equiv 9815310 \pmod n$，因而，由勒让德的方法可求出 n 的因子：$(9815310 - 2^9\cdot 31\cdot 71\cdot 97,\ 12007001) = 3001$，即 $3001 \mid 12007001$。

从上例可见，要确定哪些 $(-1)^{m+1}Q_{m+1}$ 入选，先要将 $(-1)^{m+1}Q_{m+1}$ 在某个确定的素因子（包括 -1）集上分解，再将这些分解了的 $(-1)^{m+1}Q_{m+1}$ 拼凑。这里，在某个集上分解是指在这个集中的因子全部分解出来。我们称上面那个确定的素因子（包括 -1）集为分解基集。因为，若 $p \mid (-1)^{m+1}Q_{m+1}$，则 $A_m^2 - knB_m^2 \equiv (-1)^{m+1}Q_{m+1} \equiv 0 \pmod p$，即 $\left(\frac{kn}{p}\right) = 1$ 或 0。因此，分解基集中包括的素数 p 应该使得 $\left(\frac{kn}{p}\right) = 0$ 或 1。另外，因为对诸 Q_{m+1}（尽管大多数 Q_{m+1} 不会超过 $\sqrt{kn}$）的分解仍然是件麻烦的事情。因而，分解基集中的素数不宜取得太大太多。通常，按需要，指定一个素数 p_m，

使得分解基集中的素数小于 p_m。有时，为了加快算法执行的速度或为了更有可能找到合适的一组二次剩余。分解基集中也可以包括一些大于 p_m 但小于p_m^2 的素数。

取定一个分解基集，若一个 $(-1)^{m+1}Q_{m+1}$ 的素因子全在这个分解基集中，则称它在这个分解基集上可完全分解，也称它为（对这个基集的）B_- 数。设分解基集的元素是 $p_0=-1$ 和一系列素数 $p_1, p_2, \cdots, p_h$，这时，每一个 B_- 数可以表示为一个二元数组（即 Z_2^{h+1} 中的元）。若 $(-1)^{m+1}Q_{m+1}=p_0^{u_0}p_1^{u_1}\cdots p_h^{u_h}$，则表 $(-1)^{m+1}Q_{m+1}$ 为 $(e(u_0), e(u_1), \cdots\cdots, e(u_h))$，其中 $e(u)=\begin{cases}0, u \text{ 为偶数} \\ 1, u \text{ 为奇数}\end{cases}$，这时，两个 B_- 数之积的表示就是两个 B_- 数的表示之和。若干个 B_- 数之积为平方数等价于说它们对应的表示之和为 $(0, 0, \cdots, 0)$若有 g 个（在实际应用中，通常取 $g=h+2$）B_- 数，设为 $c_1, \cdots c_g$，则可产生一个二元数组成的矩阵，这个矩阵的第 i 行为 c_i 的二元数组表示，因而，这个矩阵有 g 行，$h+1$ 列。当$g>h+1$ 时，其行向量（诸 c_i 的表示！）就必然对Z_2线性相关。因此，有若干个行使其行向量之和为$(0, 0, \cdots, 0)$（注意到 Z_2 中唯一不为零的元只有 1）这样一来，这若干个行对

应的 B_- 数之积就是一个完全平方数。因而，第二个问题就得以解决了。

在连分数算法中，计算量主要是在分解诸 $(-1)^{m+1}Q_{m+1}$ 上。计算机往往在不太可能在基集上分解的 $(-1)^{m+1}Q_{m+1}$ 上耗费很多运算。波门伦斯对此问题作了研究后，他找出了一个方法，用此方法可以排除那些不太可能在基集上分解的 $(-1)^{m+1}Q_{m+1}$。他的方法更进一步提高了连分数法的速度。

有时，在给定的基集上，很多 $(-1)^{m+1}Q_{m+1}$ 不太可能被分解以致于能分解的 $(-1)^{m+1}Q_{m+1}$ 不够用，波门伦斯的方法可以让计算机尽早回头扩充基集。

有时会发生这样的情况：尽管我们由一组 B_- 数乘起来得到了一个完全平方数，但它却导出二次同余式的一个平凡解。这时，我们可以找另外一组 B_- 数来试验，若仍然屡次失败，则可以换或扩充分解基集以得到更多的、更合适的 B_- 数，或换另外的 k 值再重复试验，直至成功。

莫利桑用连分数得到的 F_7 分解式是

$$F_7 = 59649589127497 21 \cdot 5704689200685129054721$$

最后提一下，波门伦斯对连分数方法作了一

个分析，他证明了，连分数算法的平均渐近工作量是 $O(n^{\sqrt{(1.5)(\log_2\log_2 n)/\log_2 n}})$。这个结果是从较高深的算法分析中得到的，这里不多介绍。

4. 二次筛法

二次筛法与连分数法是基于同样的想法和思路，只有两个差别，下面对此讲述。

首先，二次筛法不是用连分数产生二次剩余。而是直接产生二次剩余。设 $m=[\sqrt{n}]$，则 $Q(x)=(x+m)^2-n, x=0, \pm 1, \pm 2, \cdots$ 就是一系列模 n 的二次剩余，并且诸 $Q(x) \bmod n$ 不太大。

第二，给定一个分解基集，如何判别哪些 $Q(x)$ 可以在这个分解基集上分解？哪些不能？这里的方法也有别于连分数方法。

我们注意到，若 $Q(x) \equiv 0 \pmod{p^\alpha}$，即 $p^\alpha \mid Q(x)$，则 $Q(x+kp^\alpha) = (x+kp^\alpha+m)^2-n \equiv Q(x) \equiv 0 \pmod{p^\alpha}$，即 $p^\alpha \mid Q(x+kp^\alpha)$。因此，只要确定一个 x 使得 $p^\alpha \mid Q(x)$，就可以得到其它的一系列 $Q(x)$ 使其含有 p^α 的因子。如何确定 x 使 $p^\alpha \mid Q(x)$，这就是解同余式 $Q(x) \equiv 0 \pmod{p^\alpha}$，而这个同余的解归结为同余

式$Q(x)\equiv 0(\bmod p)$ 的解。关于素数模的二次同余式的解这个问题，勒默于1969年发表的一篇文章中仔细讨论过，得到了一个很有效的解法。读者请参看[3]。

对于某个分解基集，设它的元素是素数p_1, …, p_h，我们只要依次解同余式$Q(x)\equiv 0(\bmod p_i^{\alpha_i})$ $(i=1,\cdots,h)$然后，确定出h个等差分别为$p_1^{\alpha_1}$, …, $p_h^{\alpha_h}$的等差序列来，它们的交集仍然是个等差序列，这个序列对应的$Q(x)$就是基本上在分解基集上可以完全分解的了。因而，基本上全是B-数。这里提供了一个找B-数的方法。

波门伦斯也对二次筛法作了算法分析，他得到二次筛法的渐近计算量是

$O(n^{\sqrt{1.125\log_2\log_2 n/\log_2 n}})$，这个计算量看起来比连分数算法的计算量要小些，但这只是渐近的情况，实际上，这里的O-所示的常数比连分数算法的计算量中所示的常数要大。

用连分数方法和用二次筛法，人们都可以分解 $p\leqslant 257$ 的麦森勒数 M_p 中的合数。见附表。

5. p－1 法和 p＋1 法

波纳德首先注意到这样一个性质：若给定 n 为合数，对于n的任一个素因子p，若$p-1\mid Q$，则当 $p\nmid a$ 时，$p\mid a^Q-1$，这是由费马小定理可

知的，因而 gcd $(a^Q-1,n)>1$，若 gcd $(a^Q-1,n)\neq n$，则它就是 n 的真因子了。因而 gcd (a^Q-1,n) 就极可能产生 n 的真因子。然而，对待分解的数 n，我们不知道 n 的任何素因子，因而不知道哪个哪些 Q 合适于使 gcd (a^Q-1, n) 产生真因子。当然，我们可以对 $Q=1, 2, \cdots, \varnothing(n)$ 来试验，但这样要花费的时间就比试除法还多，因而，我们必须确定哪些 Q 最有可能使 gcd (a^Q-1,n) 产生 n 的真因子，然后对这些 Q 进行试验。

我们假设，n 含有这样一个素因子 p，它的 $p-1$ 是由比较多的小素数相乘而得的数。这时，我们可以取 Q 如下：给定一个上界 B_1（B_1 不必很大），设 β_i 是使 p_i（p_i 表示第 i 个素数）满足 $p_i^{\beta_i}\leqslant B_1$，$p_i^{\beta_i+1}>B_1$ 的数，$(i=1,2,\cdots)$，这时，可设 $Q_k=p_1^{\beta_1}\cdots p_k^{\beta_k}$（其中 p_k 是小于 B_1 的最大素数）。其次，取到 Q_k 之后，依次计算 $b_{k+1}\equiv a^{Q_{k+1}}=(a^{Q_k})^{p_{k+1}^{k+1}}\equiv b_k^{p_{k+1}^{k+1}}$ $(\bmod n)$，令 $G_k=\prod_{i=1}^{k}(b_i-1)$，$(k=1, 2, \cdots)$，然后有规则地选取 k，譬如 $k=1, 2, 4, 8, 16, 50, 100, 150, \cdots)$，计算 gcd (G_k, n)，直到得出 n 的真因子为止。

通常，若 $p-1$ 的最大素因子是 q（设 q 是

第 j 个素数，即 $q=p_j$），则计算到$k=j\approx\frac{q}{\log q}$时，便可由 gcd (G_k, n) 得到 p。有时，B_1 相当小就可以得到很大的因子，见下例。

例 分解 $10^{95}+1$。

取 $B_1=30000$ 或更小些，威廉斯用上述的 p－1 法得到了 $10^{95}+1$ 的素因子 $p=121450506296081$，其中 $p-1=2^4\cdot5\cdot13\cdot19^2\cdot15773\cdot20509$（这里人们可以看出，为什么 B_1 取30000就可以了）。威廉斯是大约计算到G_k，（$k\approx2050$）就由 gcd (G_k, n) 得到了因子 p。

另外他还得到了 $3^{136}+1$ 的素因子
$q=267009-1735108484737$，这里 $q-1=2\cdot3^2\cdot7^2\cdot19\cdot17^2\cdot569\cdot631\cdot23993$。

威廉斯通过对波纳德的 p－1 方法作细致研究后，利用努卡斯序列的一些类似于幂运算的性质，得到了一个 p＋1 法。它适合于分解这样一些合数，它们含有某些素因子 p 使得 p＋1 是由比较多的小素数相乘而得。由于，p＋1 方法牵涉到努卡斯序列的繁杂的计算，又因为 p＋1 方法的思想与 p－1 方法相同，我们这里不作介绍。

威廉斯用 p±1 法，对很多合数作了分解，他认为 p±1 方法在很多情况下比连分数方法和

二次筛法更有效。因为对于合数 n 而言，它经常含有这样一些素因子 p，使得 $p-1$ 或 $p+1$ 由较多的小素数相乘而得，所以，$p\pm1$ 方法就合适于它了。

附录：广义黎曼猜想

给定自然数 n，一个模 n 特征是指一个由整数集到复数集的函数 χ，它满足下列条件：

1. $\chi(n_1 \cdot n_2) = \chi(n_1)\chi(n_2)$，$n_1, n_2 \in Z$。

2. $\chi(n_1) = 0$ 当且仅当 $(n_1, n) > 1$。

3. 若$n_1 \equiv n_2 (\mathrm{mod}\, n)$，则 $\chi(n_1) = \chi(n_2)$。

给定自然数 n 及模 n 的一个特征χ，定义函数 $L(\chi, s) = \sum_{k=1}^{\infty} \frac{\chi(k)}{k^s}$，其中 s 是复数，其实部大于1。由复函数理论，$L(\chi, s)$ 可以解析开拓到对实部大于零的 s 都有定义。

广义黎曼猜想（ERH） 对任何自然数 n 及其特征 χ，$L(\chi, s)$ 的零点落在 $\mathrm{Re}\, s = \frac{1}{2}$这根直线上。这里 Res 表示 s 的实部。

ERH 是世界著名难题，至今还没解决。

参考文献

〔1〕 柯召、孙琦，数论讲义（上册），高等教育出版社，1986.

〔2〕 Knuth, The art of Computer programming, Vol. 2, 1981.

〔3〕 Lehmer, D. H., Computer technology applied to the theory of numbers, in Studies Number Theory, 117—151, (1969).

〔4〕 Williams, H. C., Primality testing on a Computer, Ars. Combin. 5 (1978), 127—185.

〔5〕 Dixon. J. D. Factorization and primality tests, Amer. Math. Monthly, 6 (1984), 333—352.

注 〔5〕是一篇内容丰富的综述文章，附有63篇参考文献.

中英文人名表

一至五画

卜利尔哈特（Brillhart）

马递佳塞维奇（Matijasevič）

瓦格斯塔夫（Wagstaff）

贝利（Baillie）

布勒恩特（Brent）

坎特伯雷（Canterbury）

加得（Judd）

艾德利曼（Adleman）

卡米歇尔（Carmichael）

六至八画

安克尼（Ankery）

麦森勒（Mersenne）

希尔伯特（Hilbert）

克卢丝（Knuth）

努卡斯（Lucas）

狄利克雷（Dirichlet）

波赫曼（Bohman）

波克林顿（Pocklington）

波门伦斯（Pomenrance）

泡尔斯（Powers）
波纳德（Pollard）

九至十一画

欧几里得（Euclid）
欧拉（Euler）
威尔逊（Wilson）
埃那托森斯（Eratosthenes）
费马（Fermat）
威廉斯（Williams）
高斯（Gauss）
莱布尼兹（Leibniz）
莫利桑（Morrison）
勒默（Lehmer）
维路（Vélu）
勒恩斯爵（Lenstra）

十二画以上

斐波那契（Fibonacci）
雅可比（Jacobi）
塞尔弗来季（Selfridge）
普罗丝（Proth）
赖宾（Rabin）
鲁梅利（Rumely）
葛莱启来（Kraitchik）
斯诺文斯基（Slowinski）

鲁宾逊（Robinson）

斯塔克（Stark）

赛路斯（Sarrus）

黎曼（Riemann）

蒙特克梅利（Montegemery）

蒙特卡罗（Monte—Carlo）

缪内（Miller）